July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 1

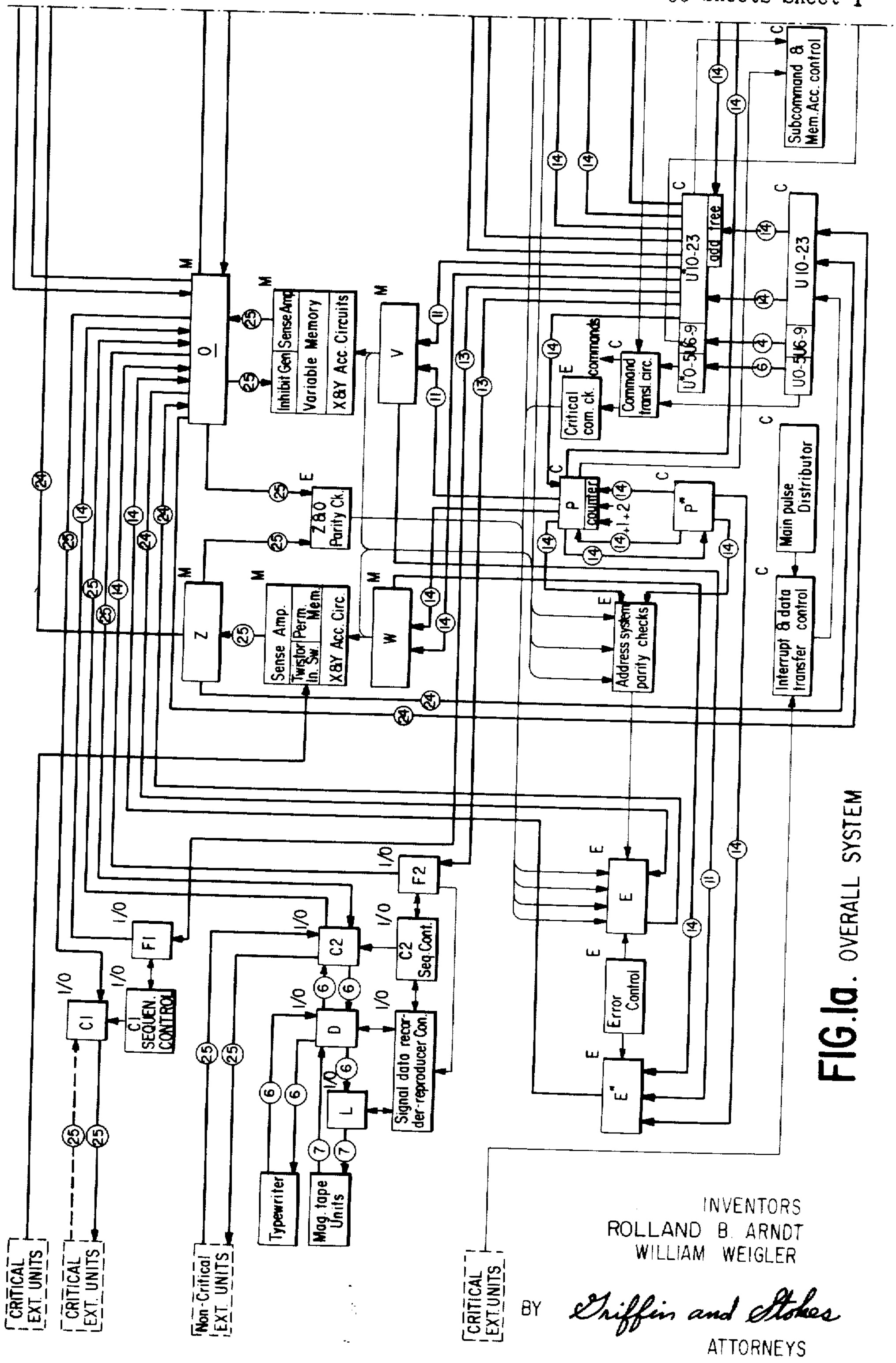


FIG. 1a. OVERALL SYSTEM

INVENTORS
ROLLAND B. ARNDT
WILLIAM WEIGLER

BY Griffin and Stokes
ATTORNEYS

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 2

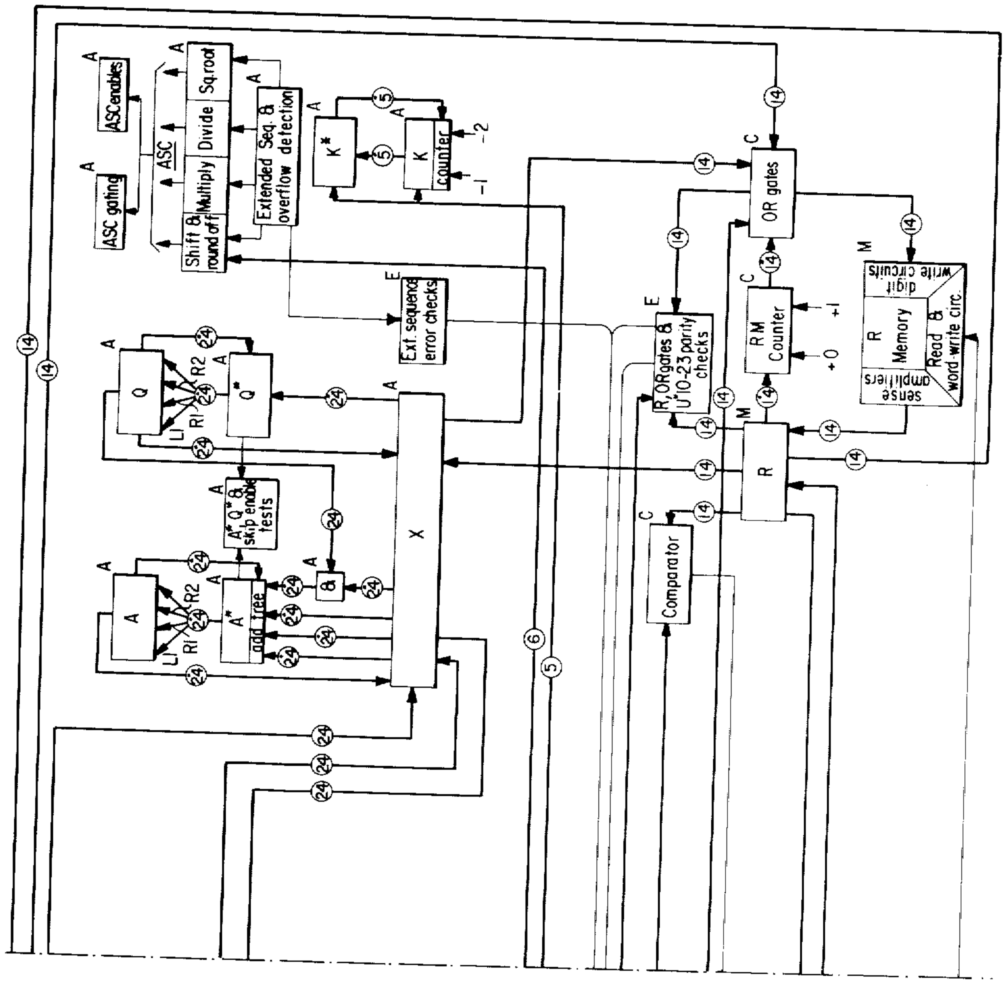


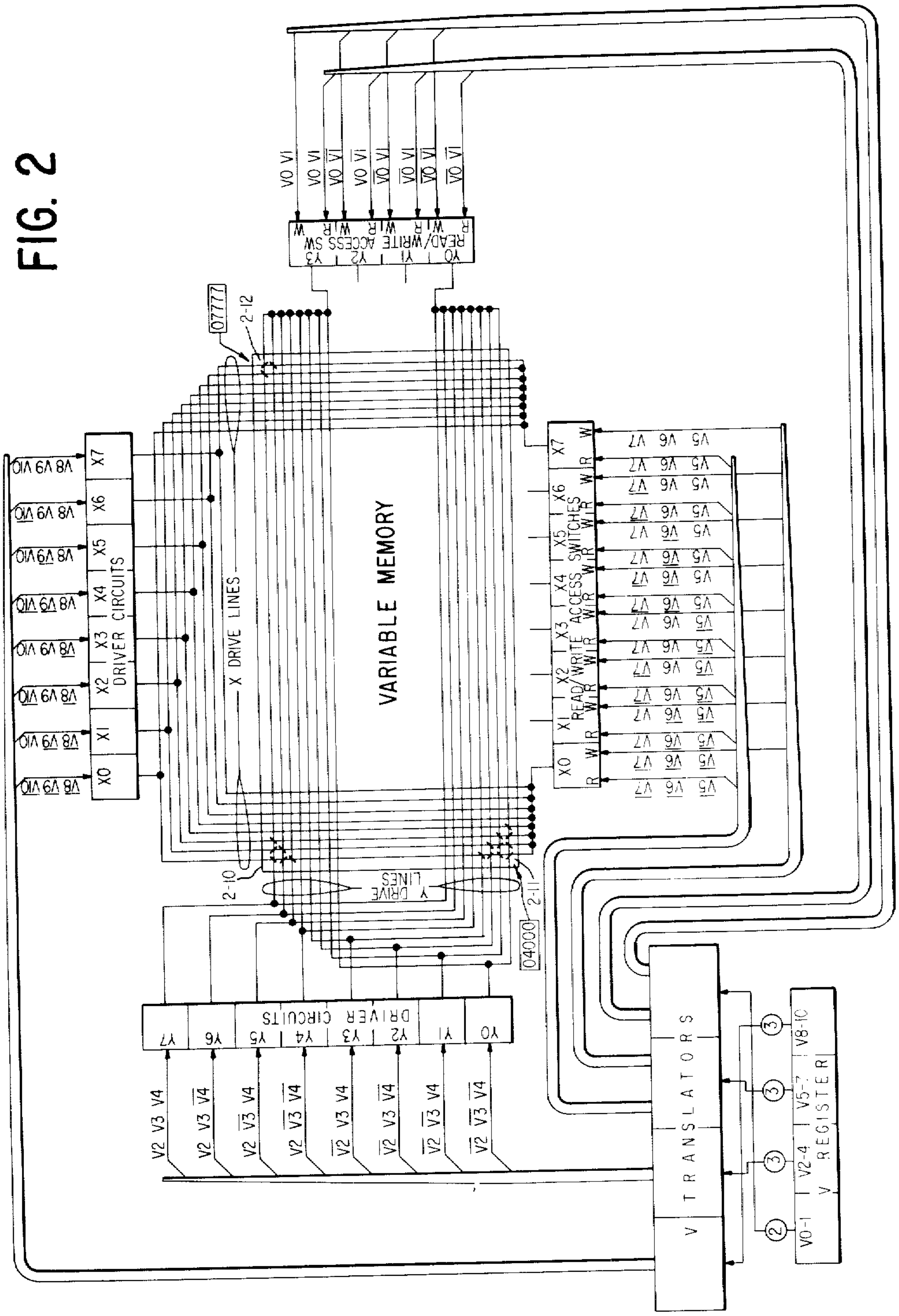
FIG. 1b
OVERALL SYSTEM

July 12, 1966 R. B. ARNDT ET AL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 3

FIG. 2



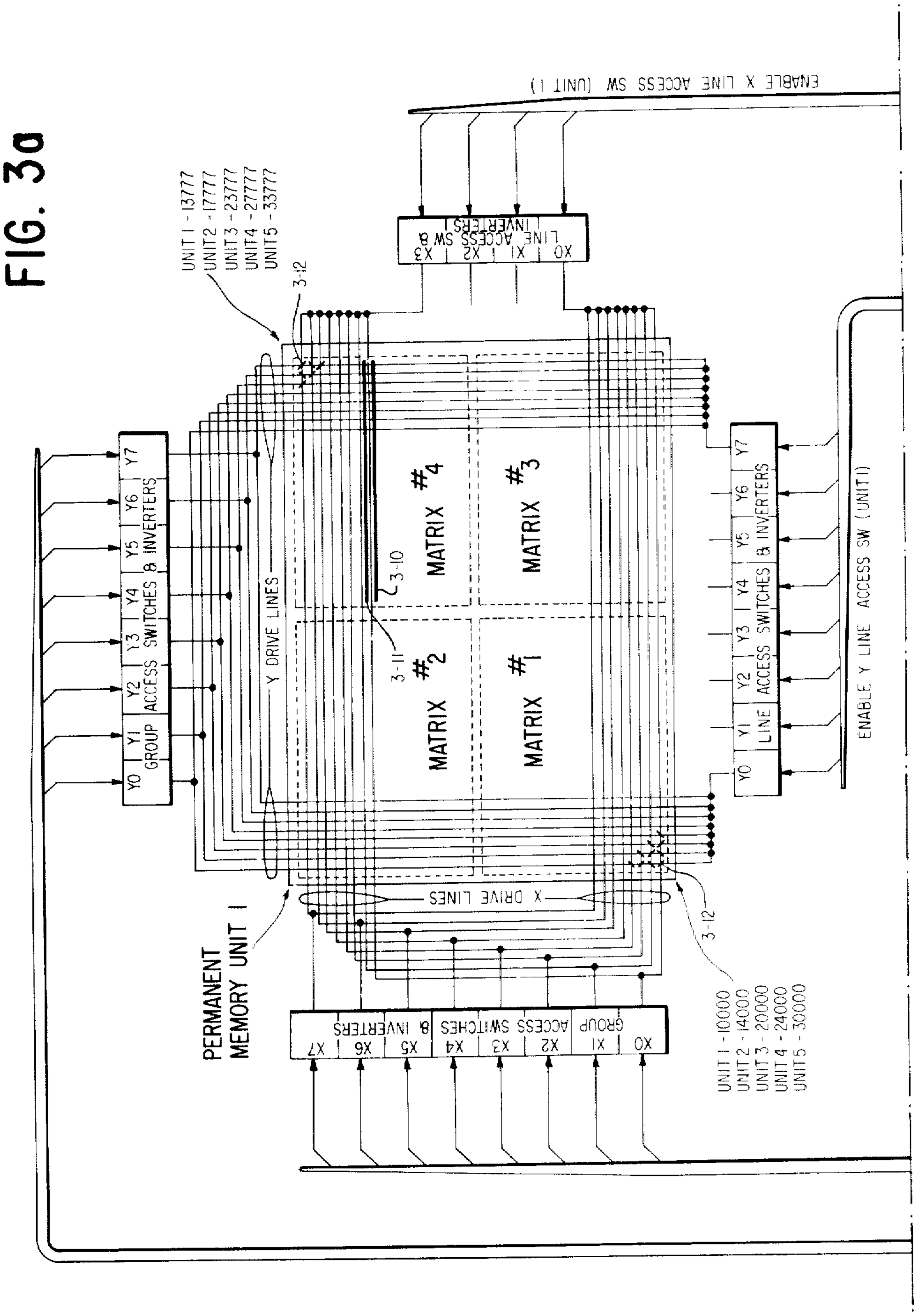
VARIABLE MEMORY
X DRIVE LINES
Y DRIVE LINES
X0 X1 X2 X3 X4 X5 X6 X7 DRIVER CIRCUITS
Y0 Y1 Y2 Y3 Y4 Y5 Y6 Y7 DRIVER CIRCUITS
X0 X1 X2 X3 X4 X5 X6 X7 READ/WRITE ACCESS SWITCHES
Y0 Y1 Y2 Y3 READ/WRITE ACCESS SW
W R
V8 V9 V10
V5 V6 V7
V2 V3 V4
V0 V1
07777
04000
2-10
2-11
2-12
V TRANSLATORS
V REGISTER
V0-1 V2-4 V5-7 V8-10
2
3

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 4

FIG. 3a


ENABLE X LINE ACCESS SW (UNIT 1)
X3 | X2 | X1 | X0 LINE ACCESS SW & INVERTERS
UNIT 1 - 13777
UNIT 2 - 17777
UNIT 3 - 23777
UNIT 4 - 27777
UNIT 5 - 33777
3-12
Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0 GROUP ACCESS SWITCHES & INVERTERS
Y DRIVE LINES
MATRIX #4
MATRIX #3
MATRIX #2
MATRIX #1
3-10
3-11
Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0 LINE ACCESS SWITCHES & INVERTERS
ENABLE Y LINE ACCESS SW (UNIT 1)
X DRIVE LINES
3-12
PERMANENT MEMORY UNIT 1
X7 X6 X5 X4 X3 X2 X1 X0 GROUP ACCESS SWITCHES & INVERTERS
UNIT 1 - 10000
UNIT 2 - 14000
UNIT 3 - 20000
UNIT 4 - 24000
UNIT 5 - 30000

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 5

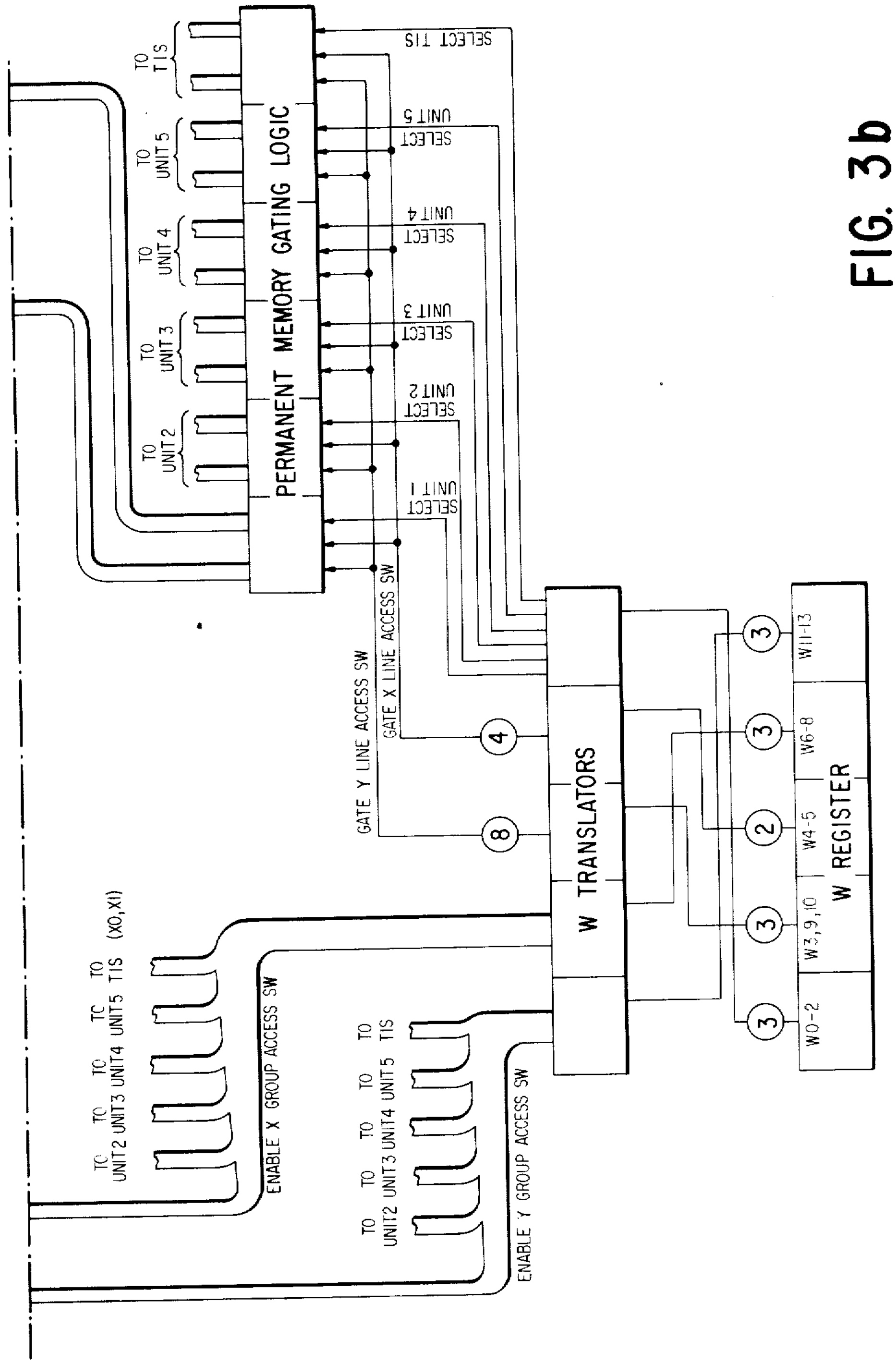


FIG. 3b

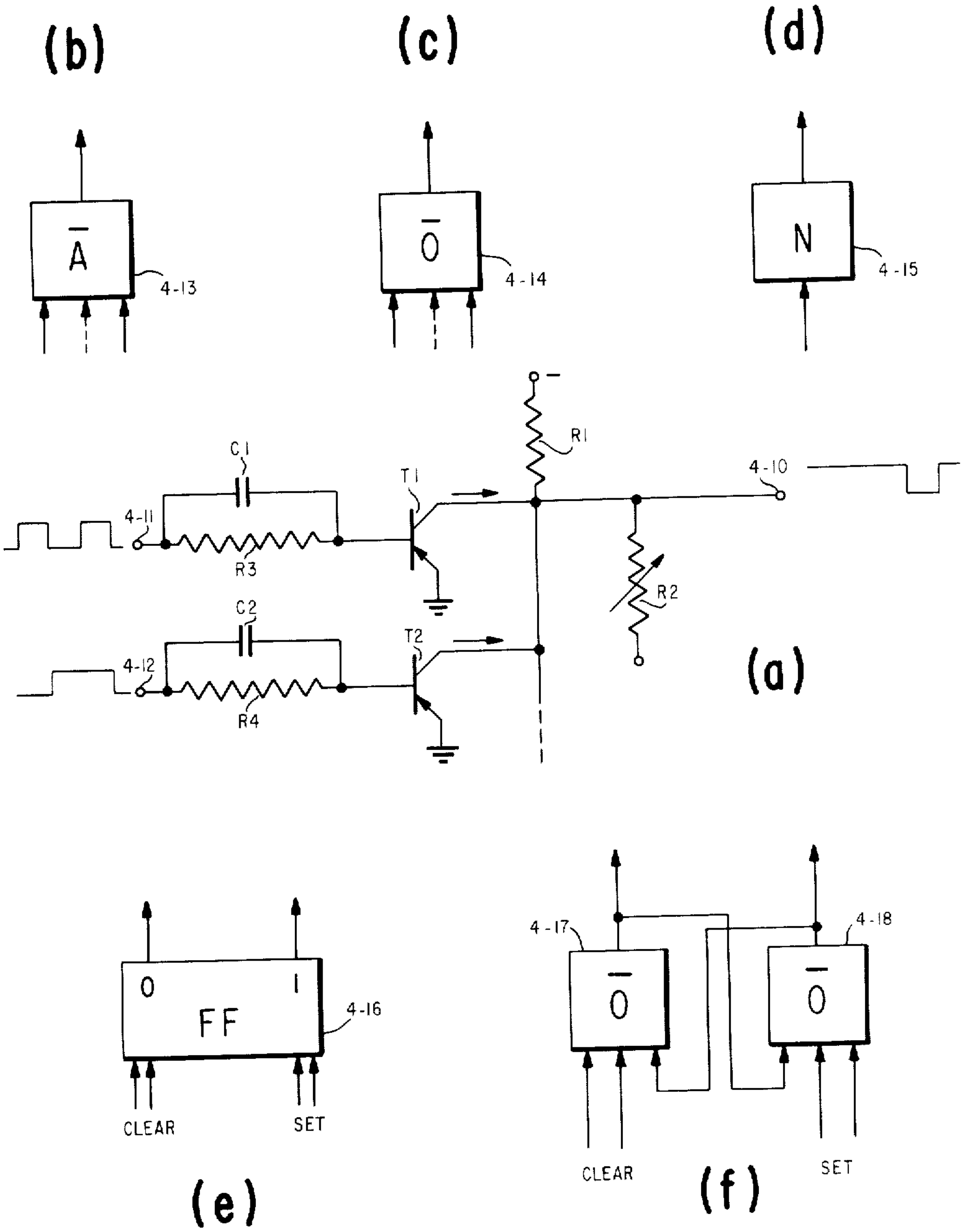

(b)
(c)
(d)
Ā
4-13
Ō
4-14
N
4-15
–
R1
C1
T1
4-10
4-11
R3
R2
C2
T2
4-12
R4
(a)
0
1
FF
4-16
CLEAR
SET
(e)
4-17
4-18
Ō
Ō
CLEAR
SET
(f)

FIG.4

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

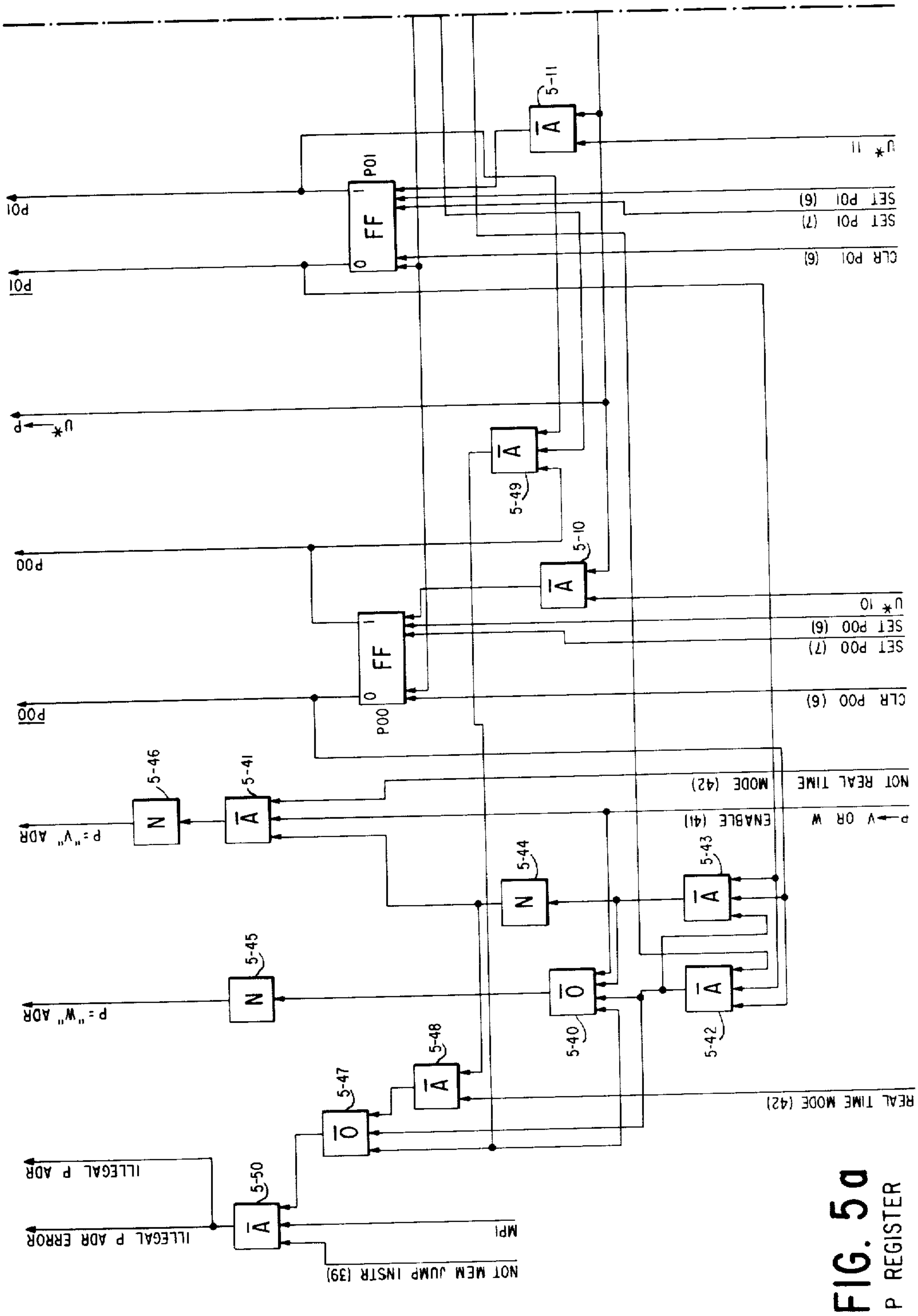


FIG. 5a

P REGISTER

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

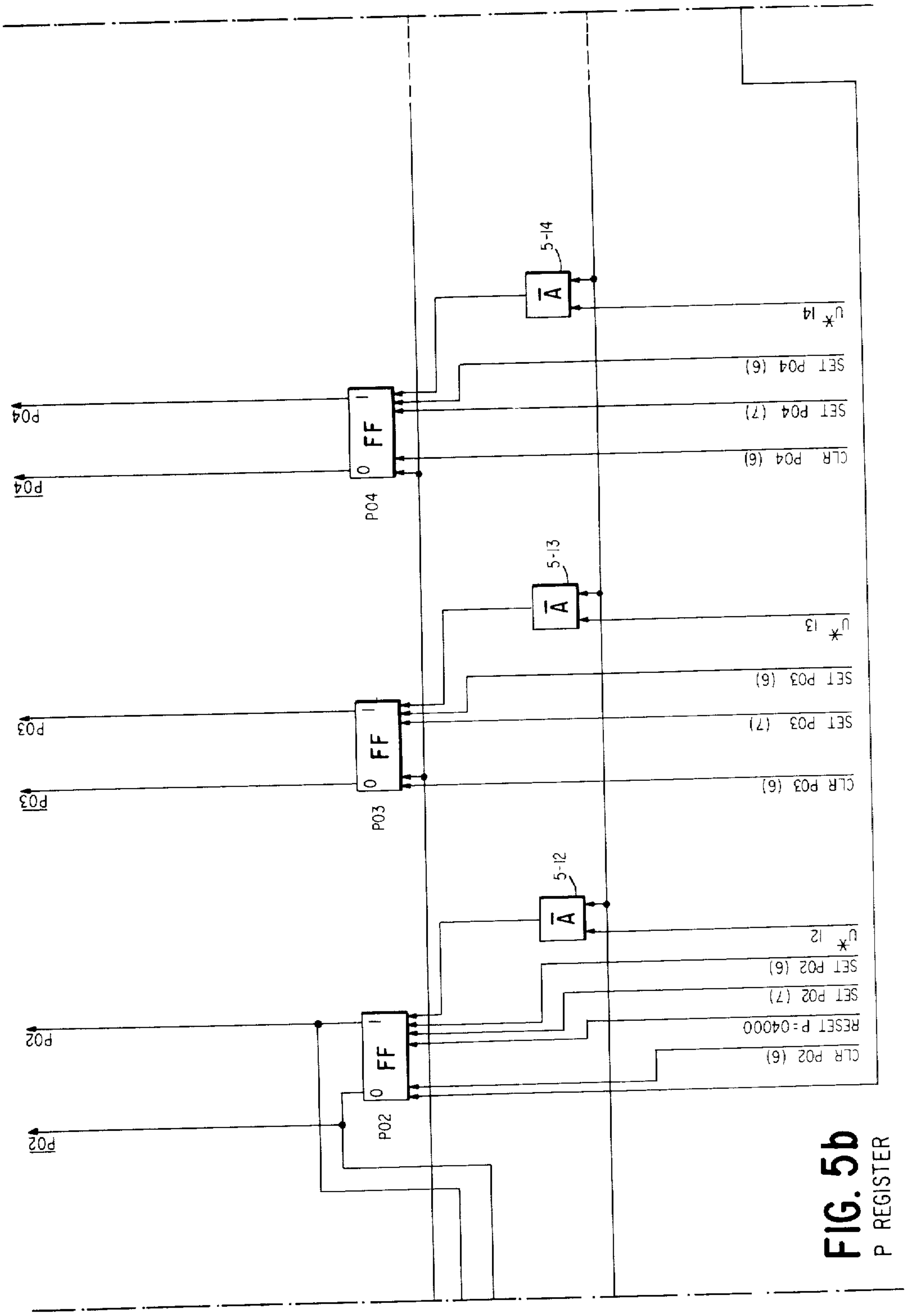


FIG. 5b

P REGISTER

July 12, 1966 R. B. ARNDT ET AL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 9

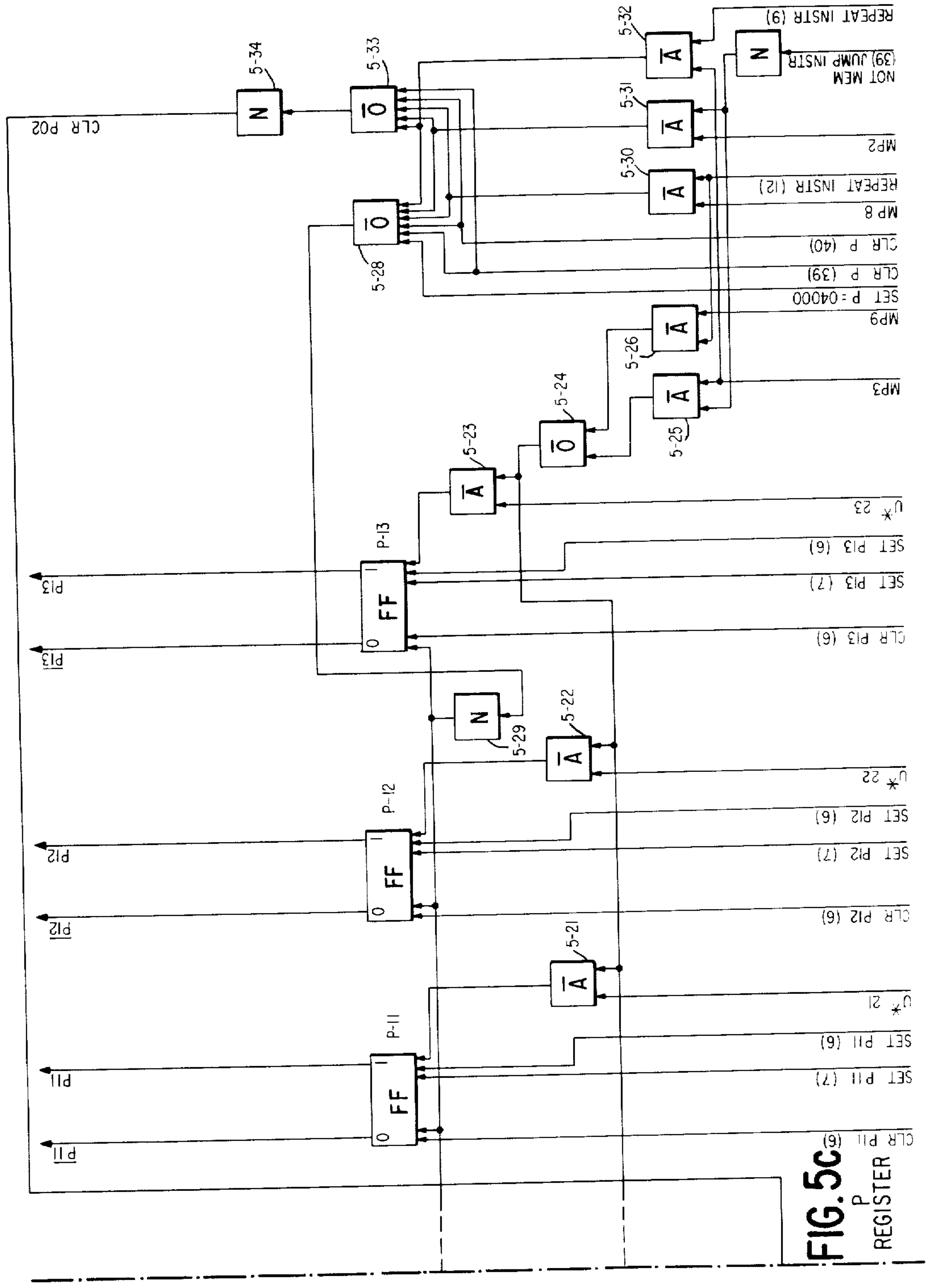


FIG. 5c
P REGISTER

July 12, 1966 R. B. ARNDT ET AL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 10

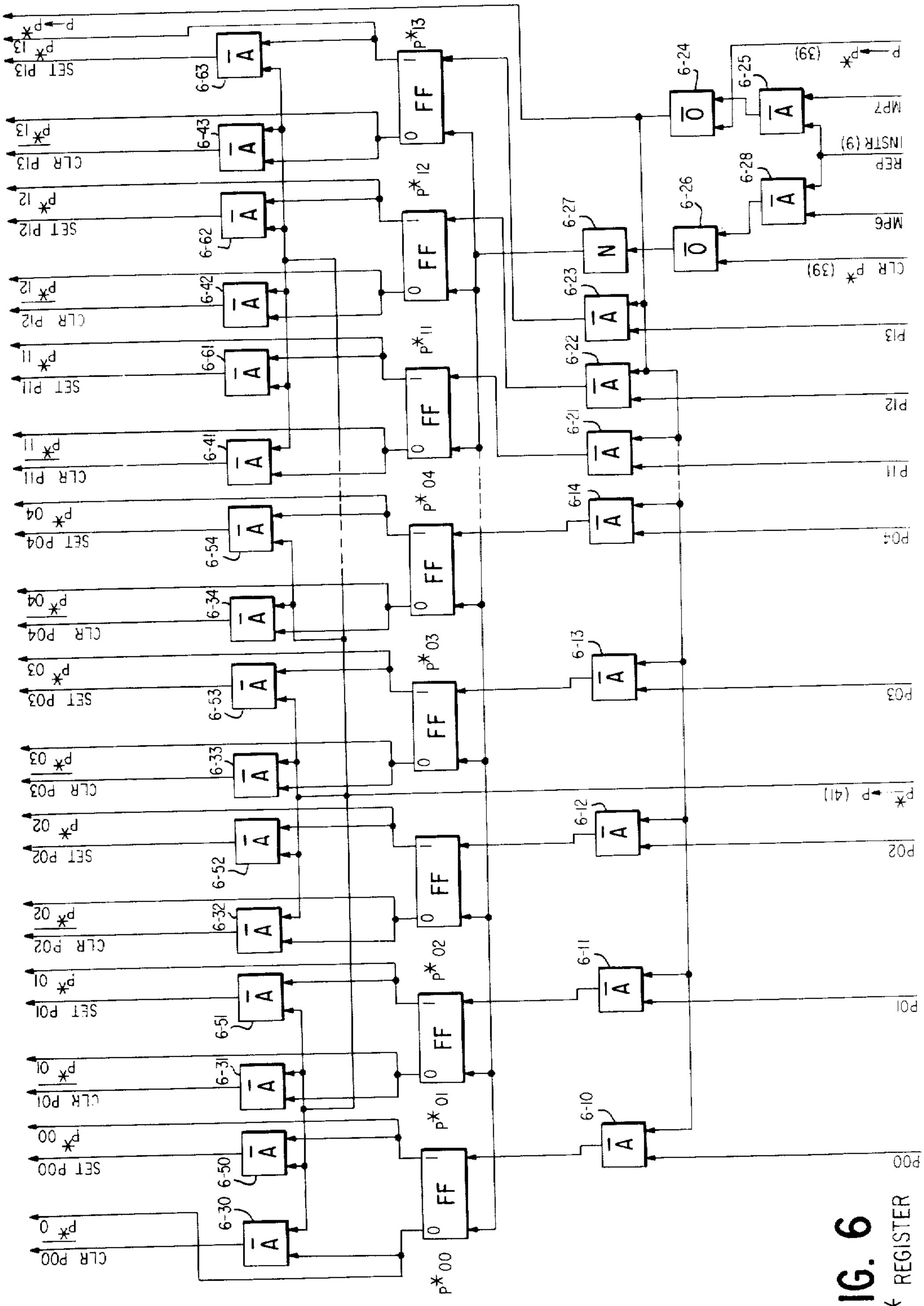


FIG. 6

P* REGISTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 11

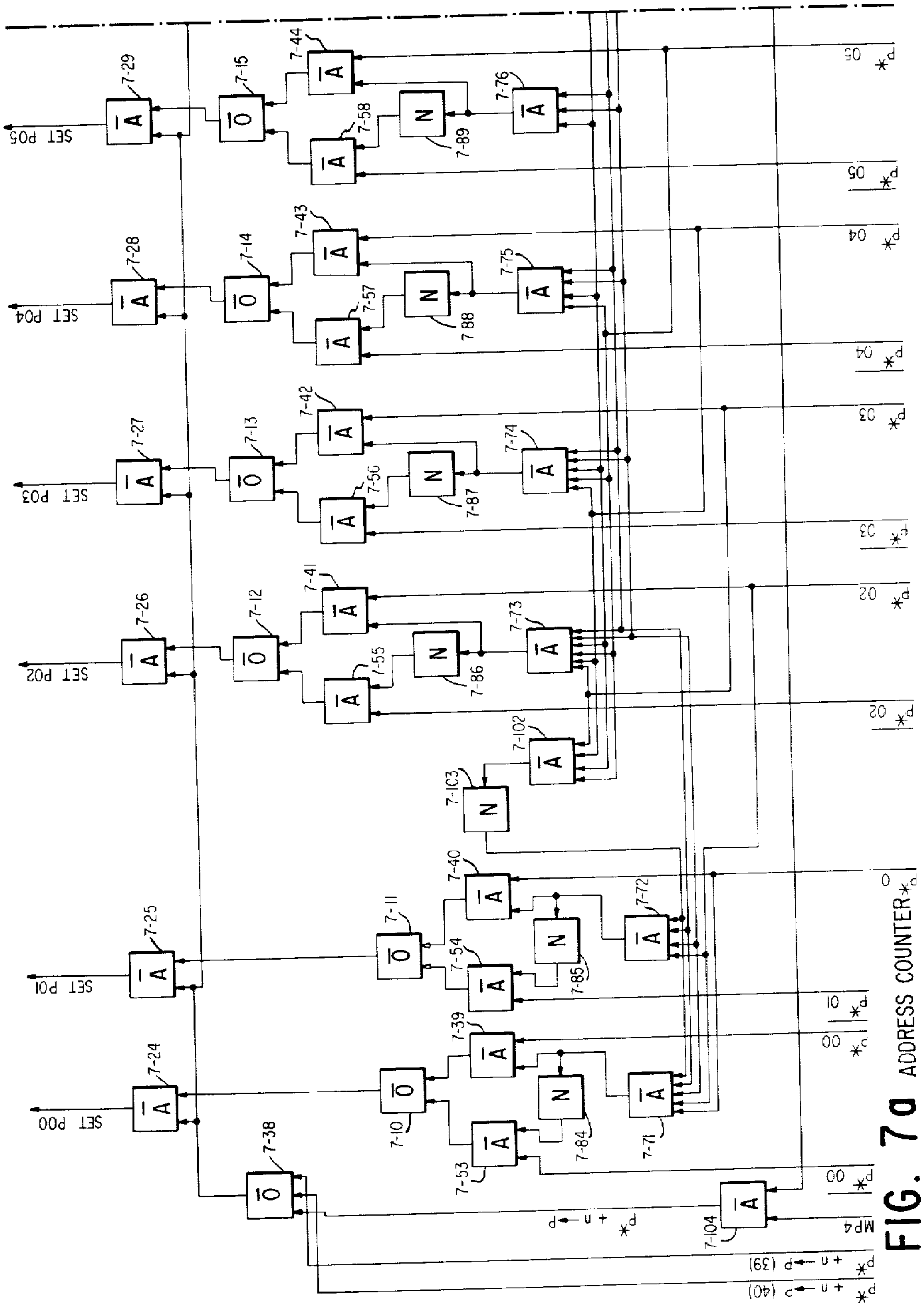


FIG. 7a ADDRESS COUNTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 12

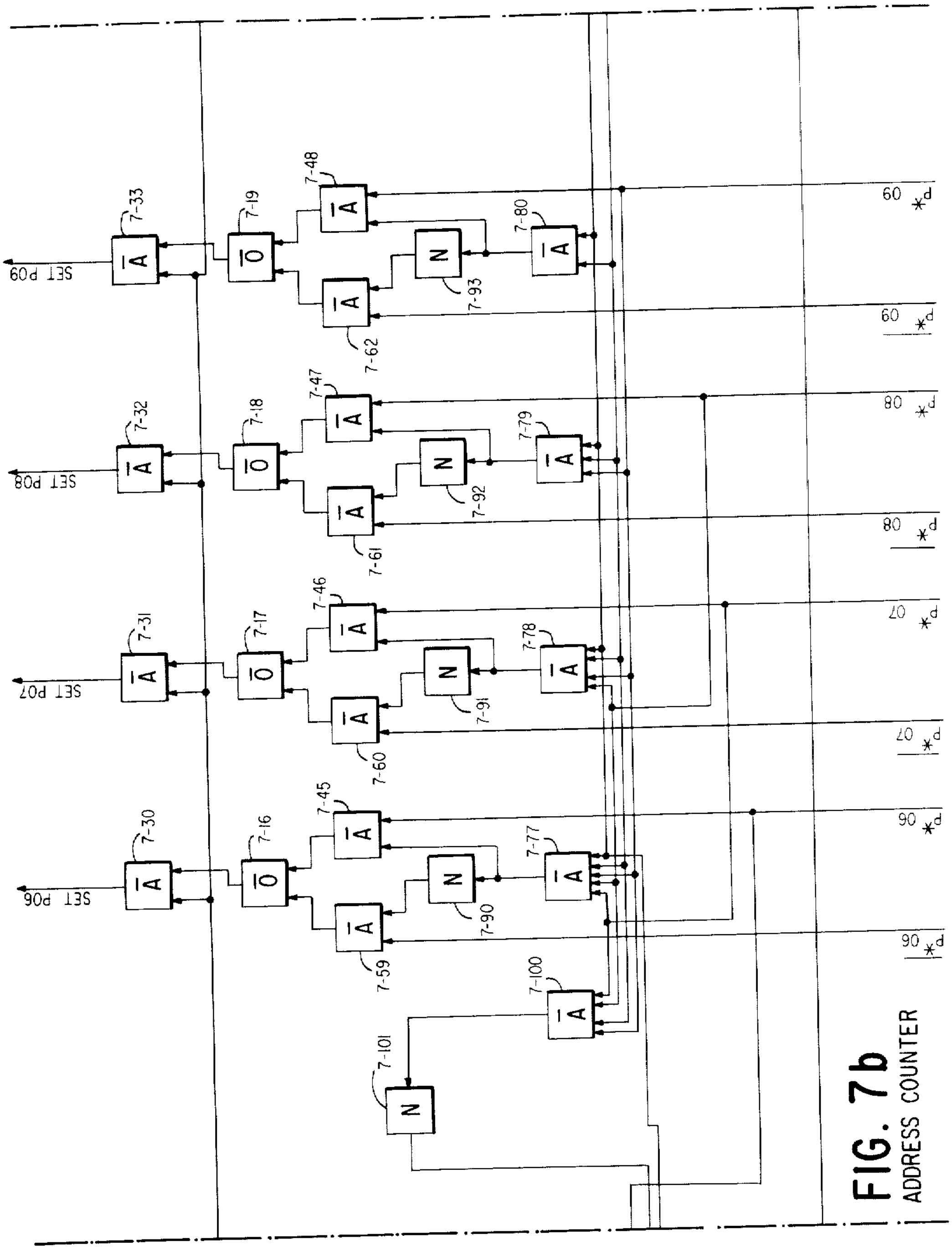


FIG. 7b
ADDRESS COUNTER

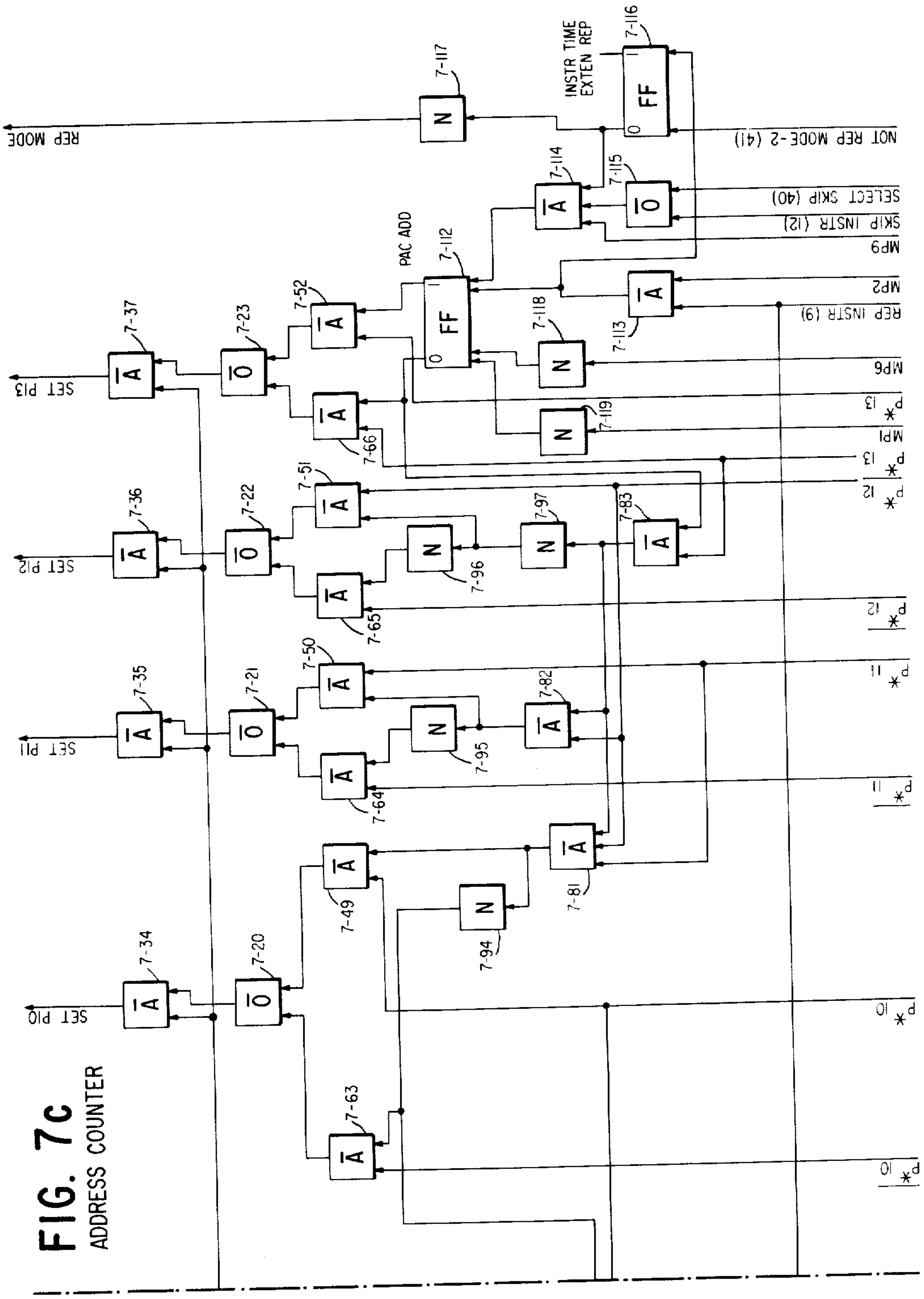


FIG. 7c
ADDRESS COUNTER

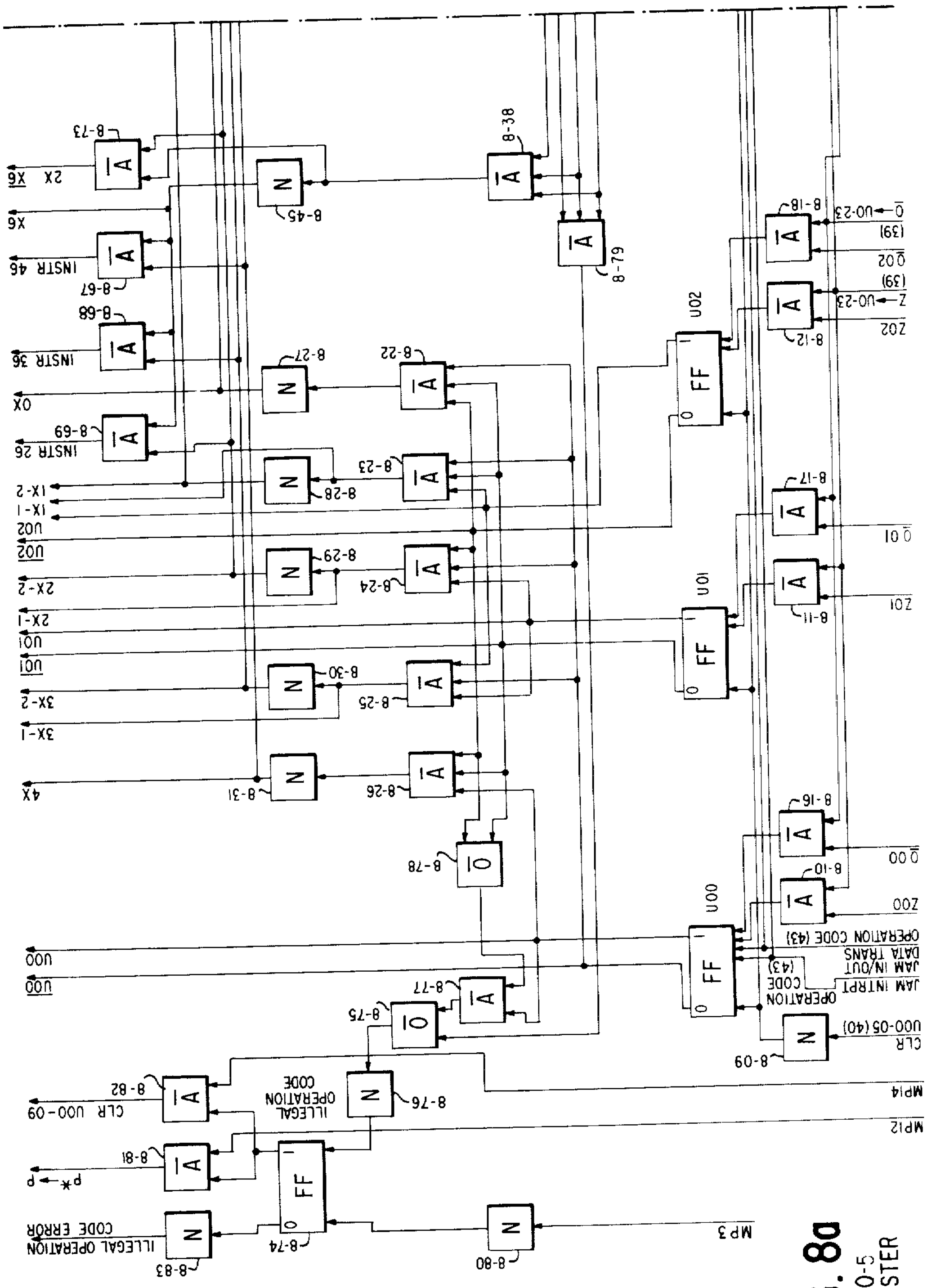


FIG. 8a
U0-5
REGISTER

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 15

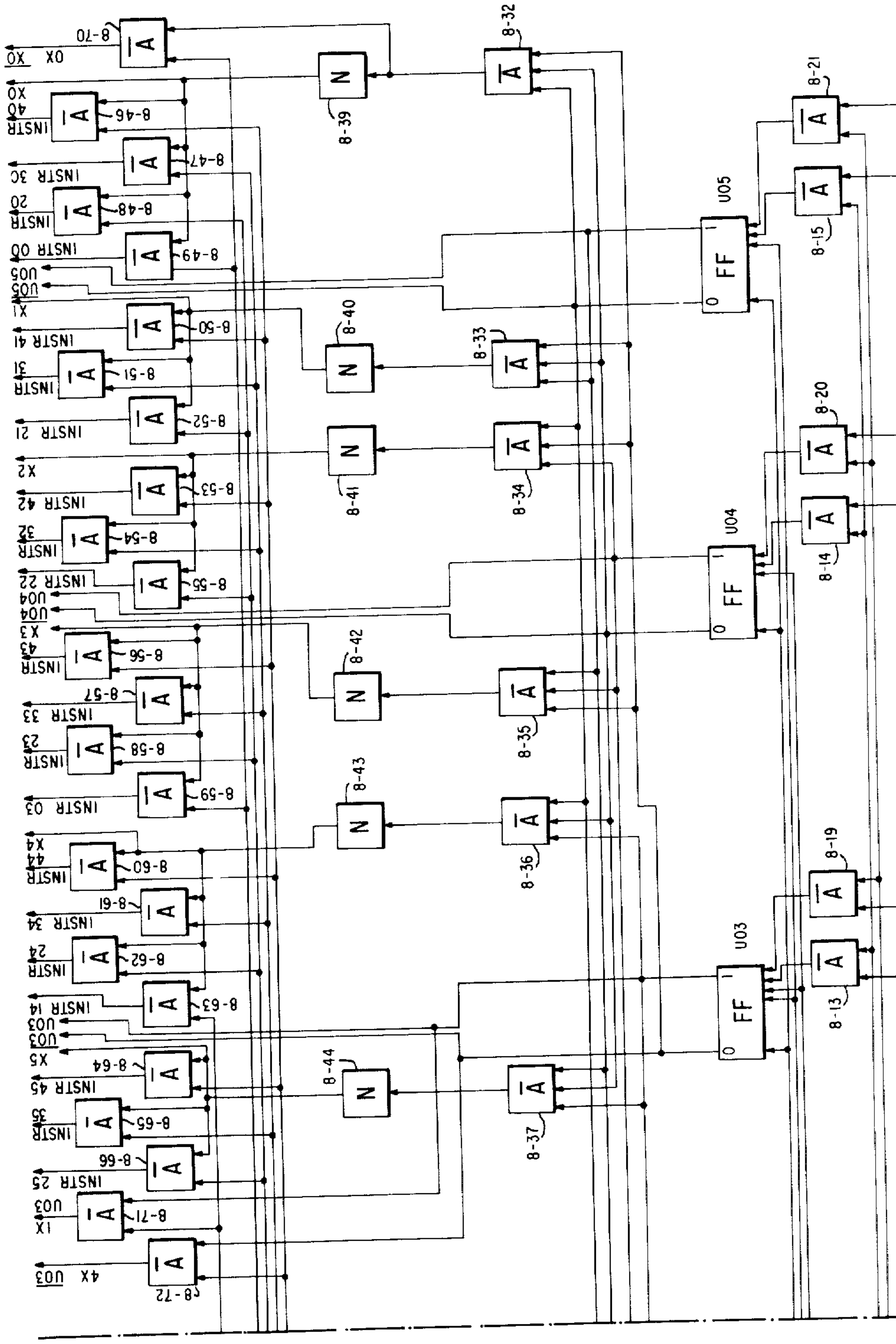


FIG. 8b UO-5 REGISTER

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 16

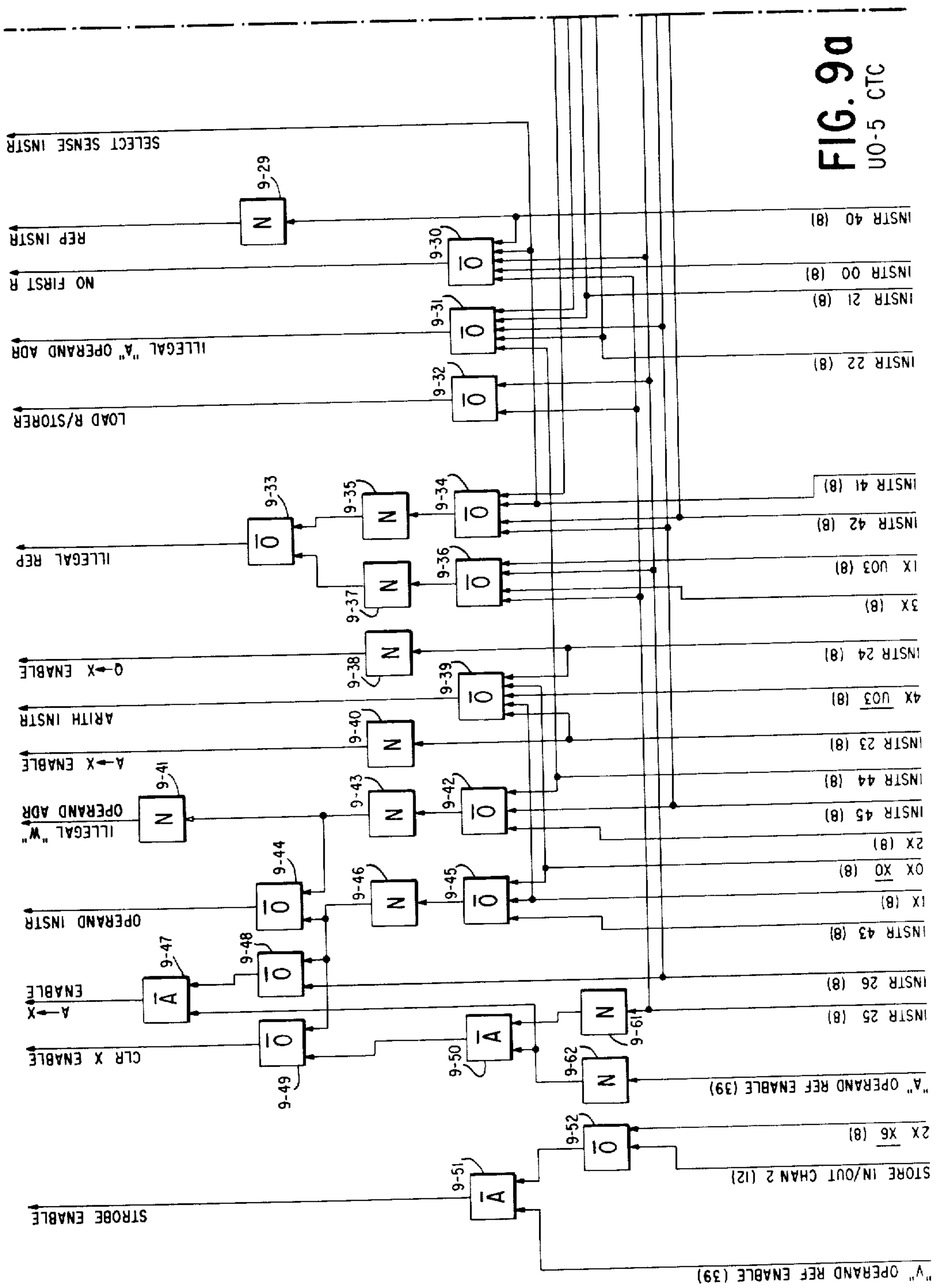


FIG. 9a

UO-5 CTC

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 17

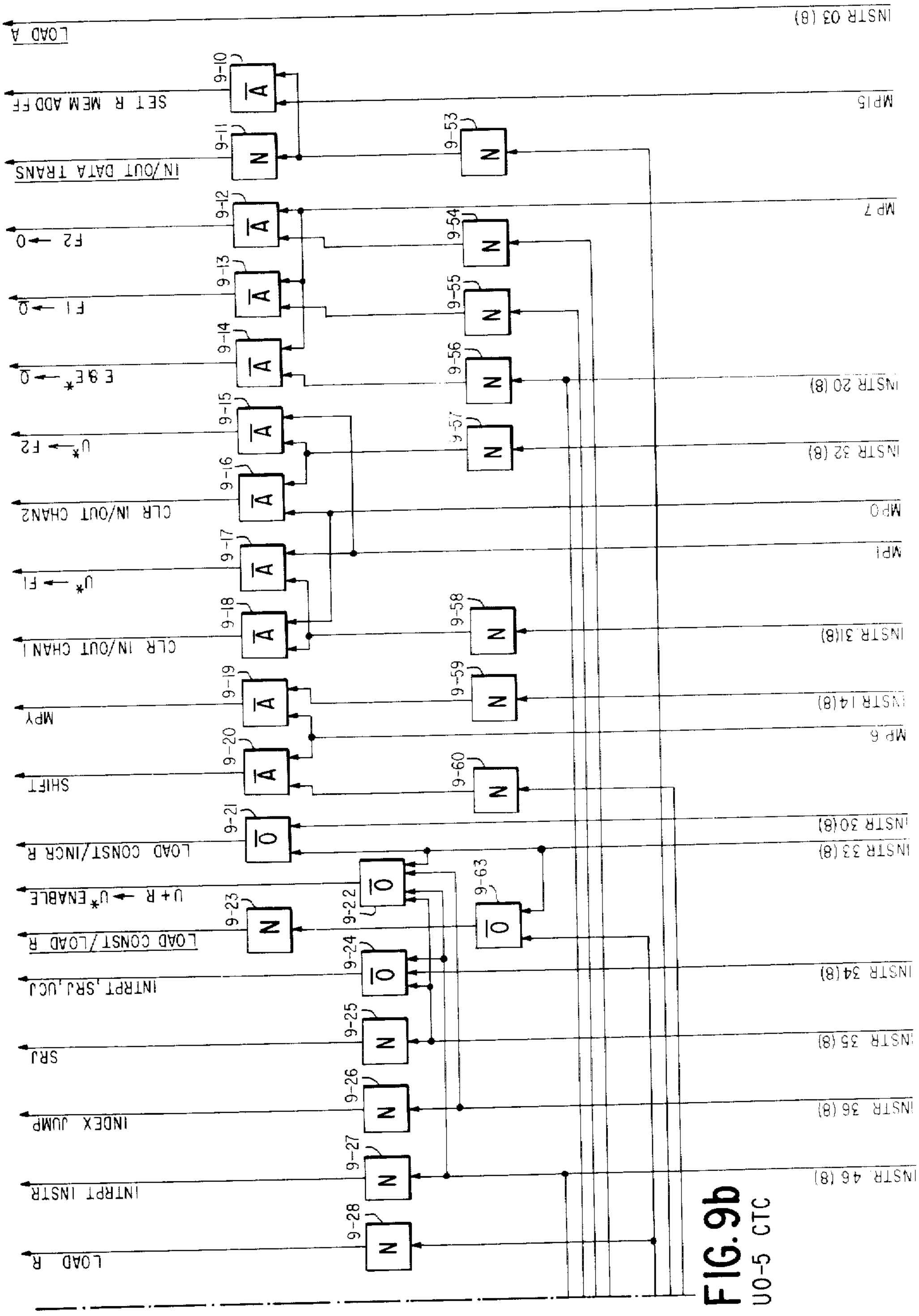


FIG. 9b
UO-5 CTC

July 12, 1966

R. B. ARNDT ETAL

3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 18

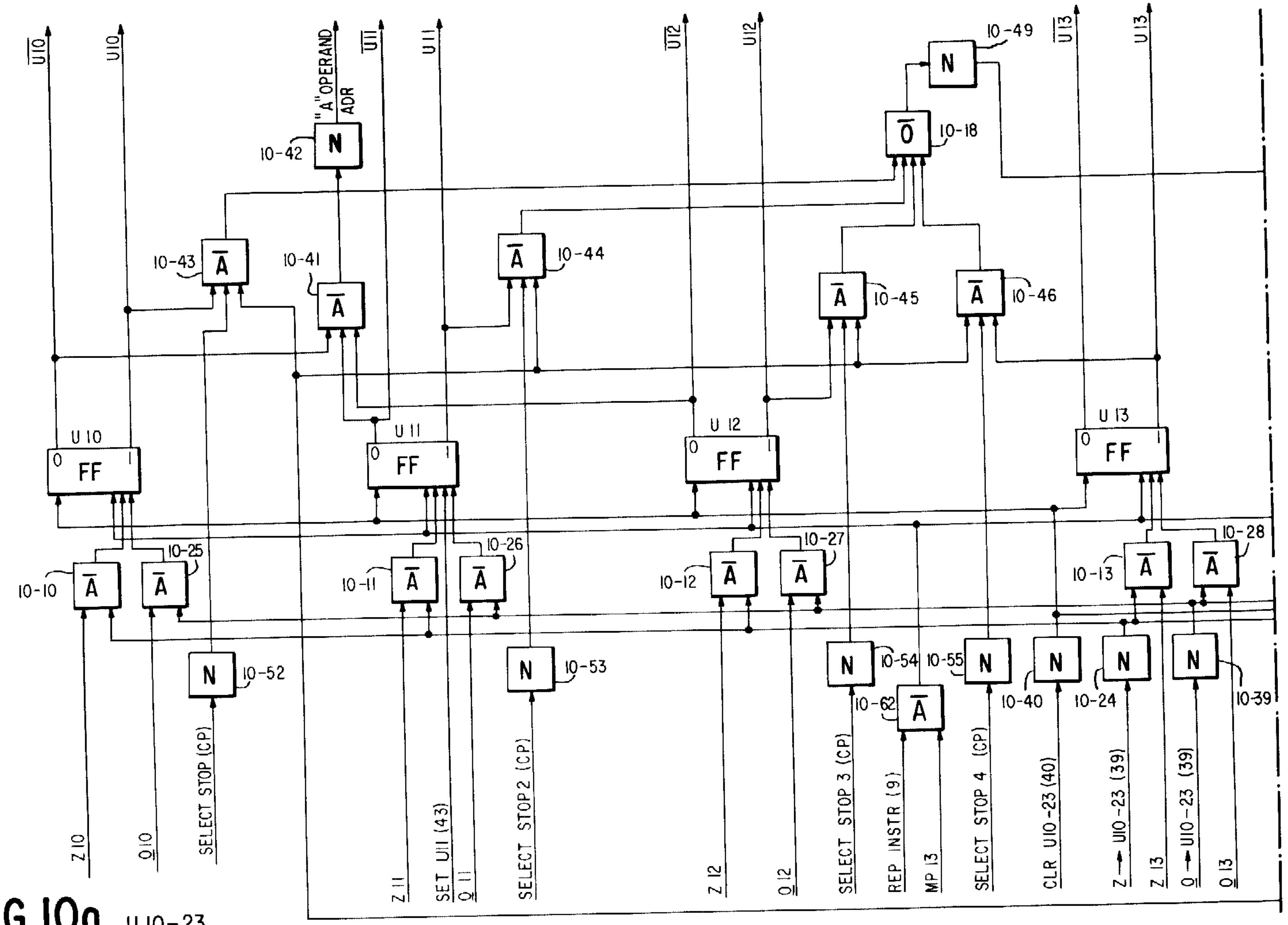


FIG. 10a U 10-23

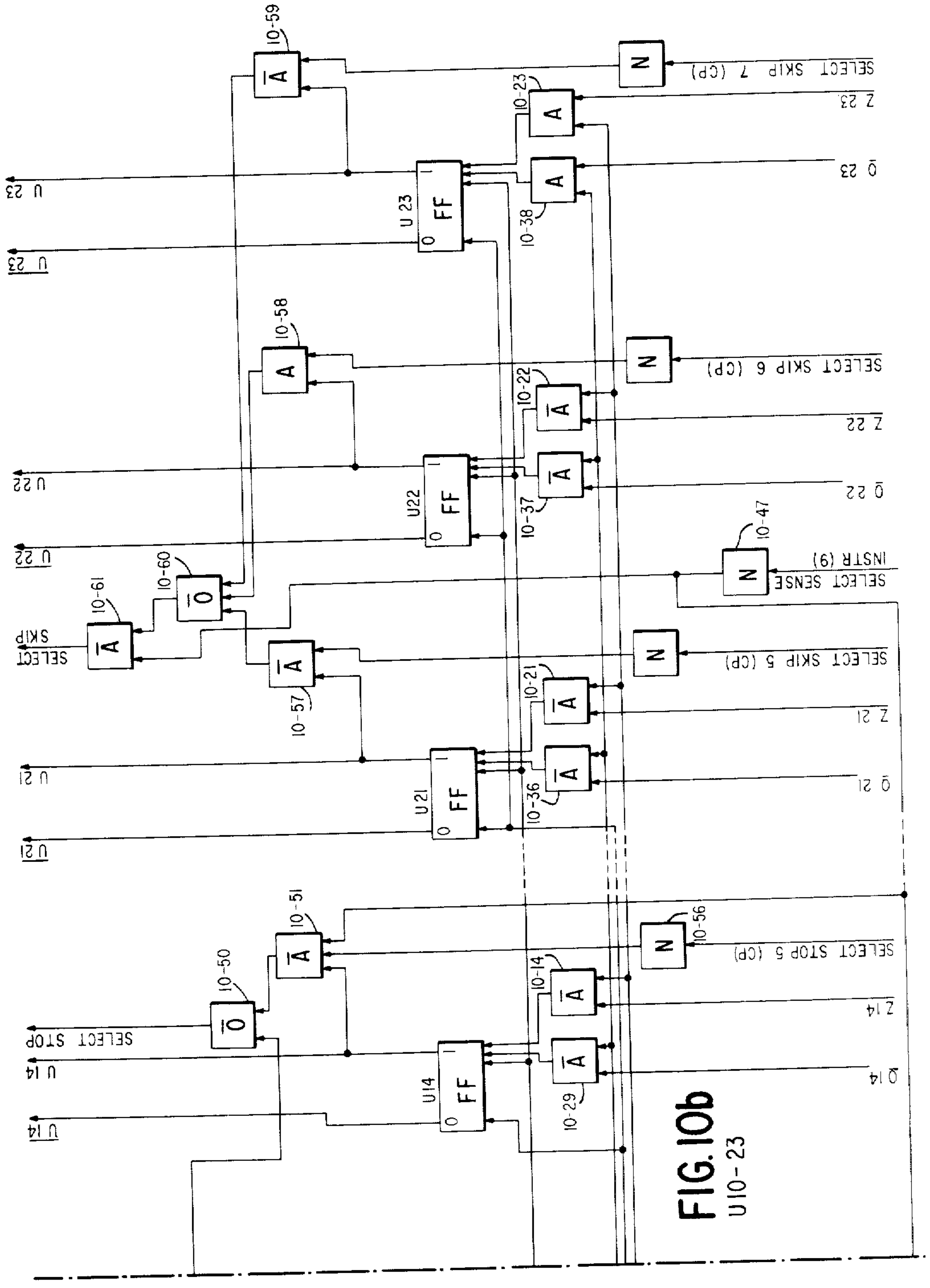


FIG. 10b
U10-23

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 20

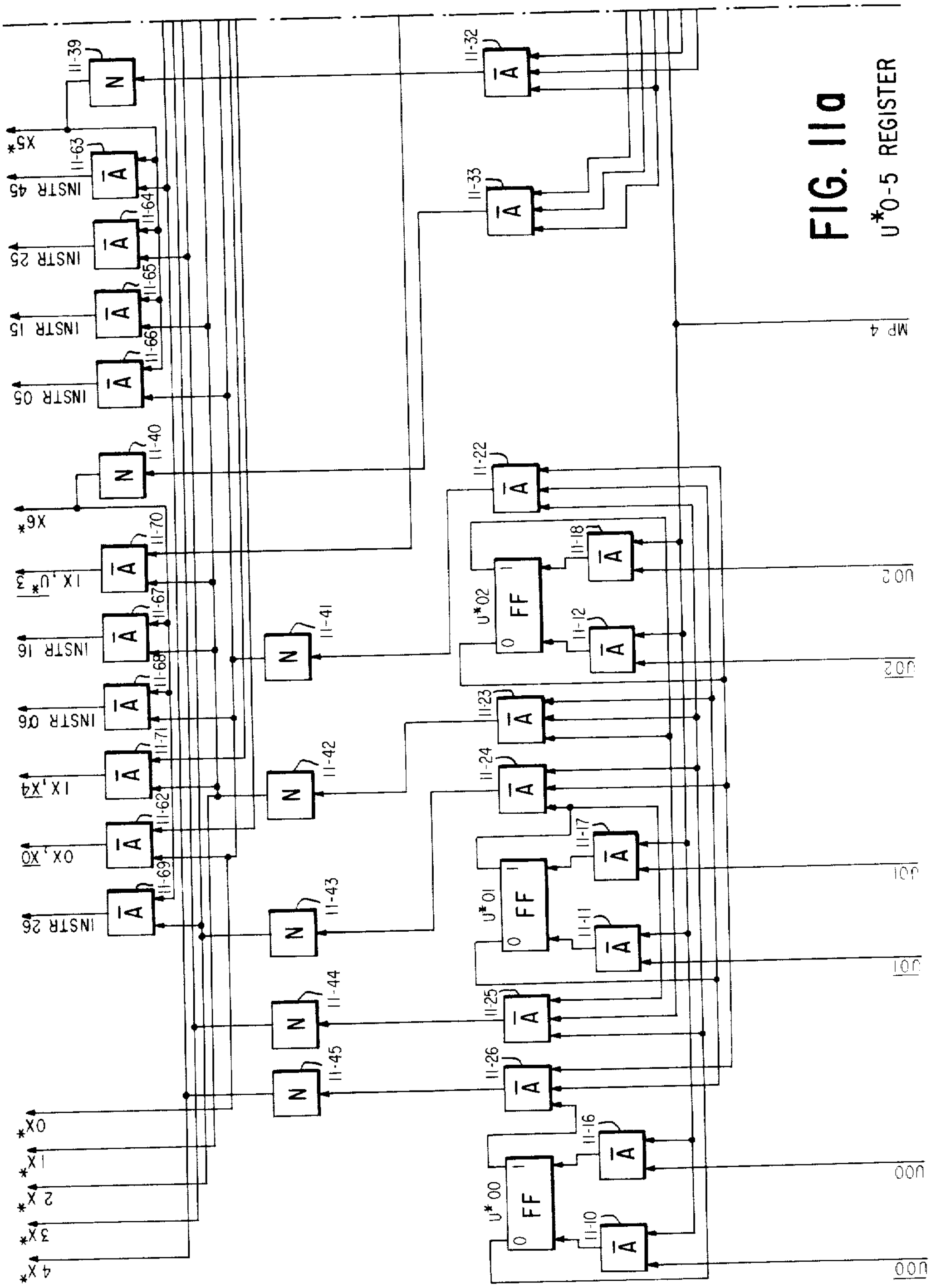


FIG. 11a

U*0-5 REGISTER

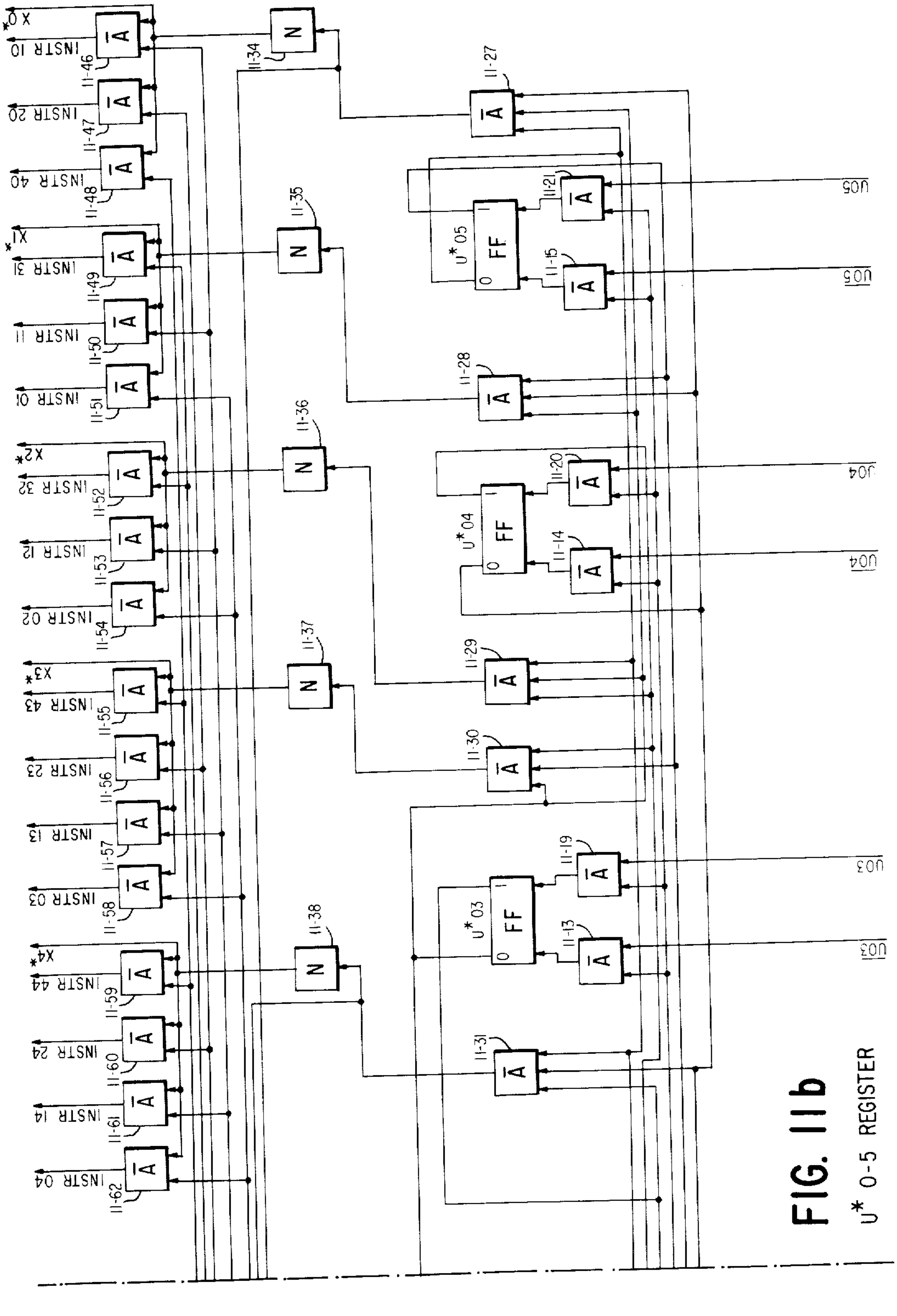


FIG. 11b

U* 0-5 REGISTER

July 12, 1966 R. B. ARNDT ET AL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 22

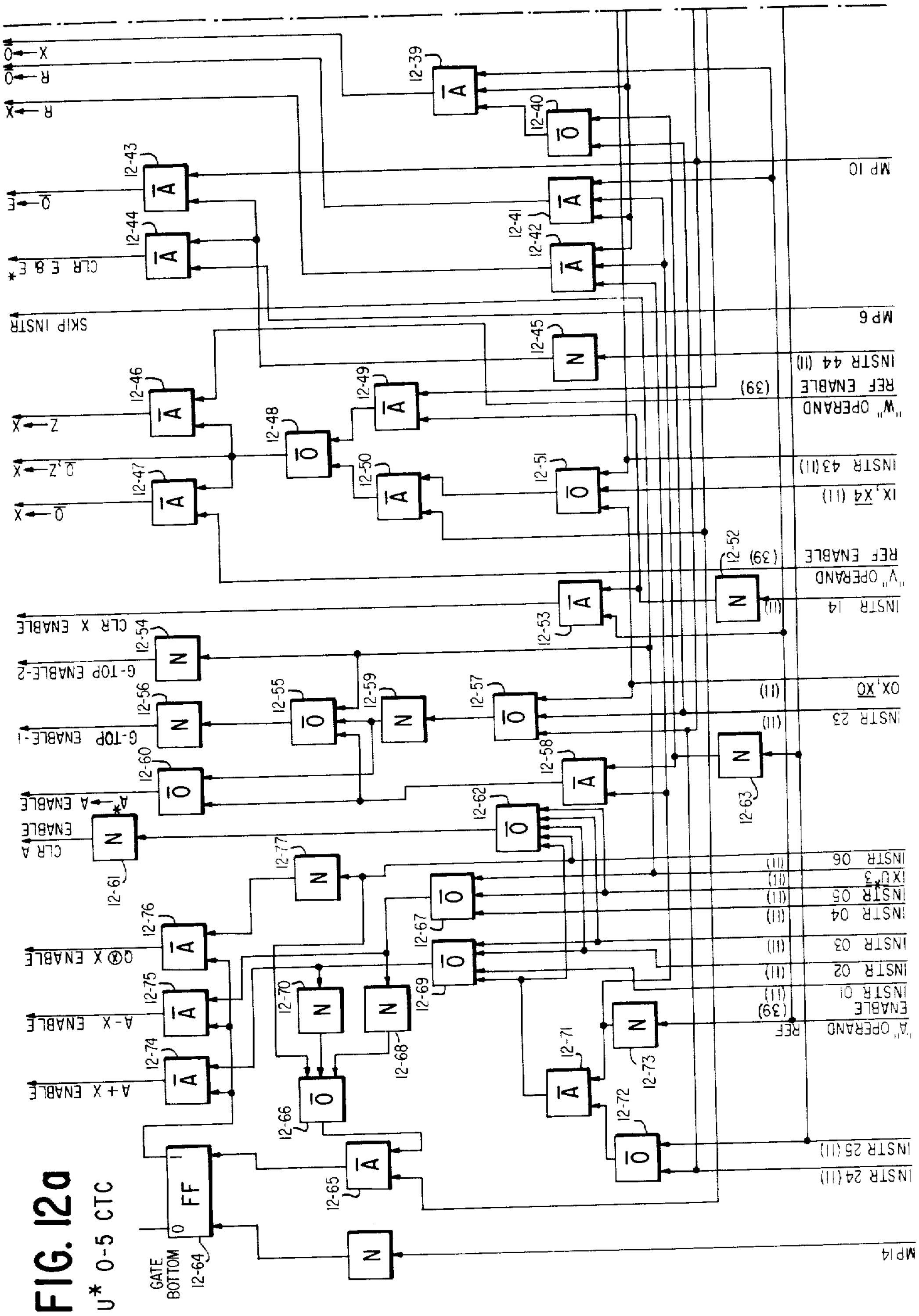


FIG. 12a
U* 0-5 CTC

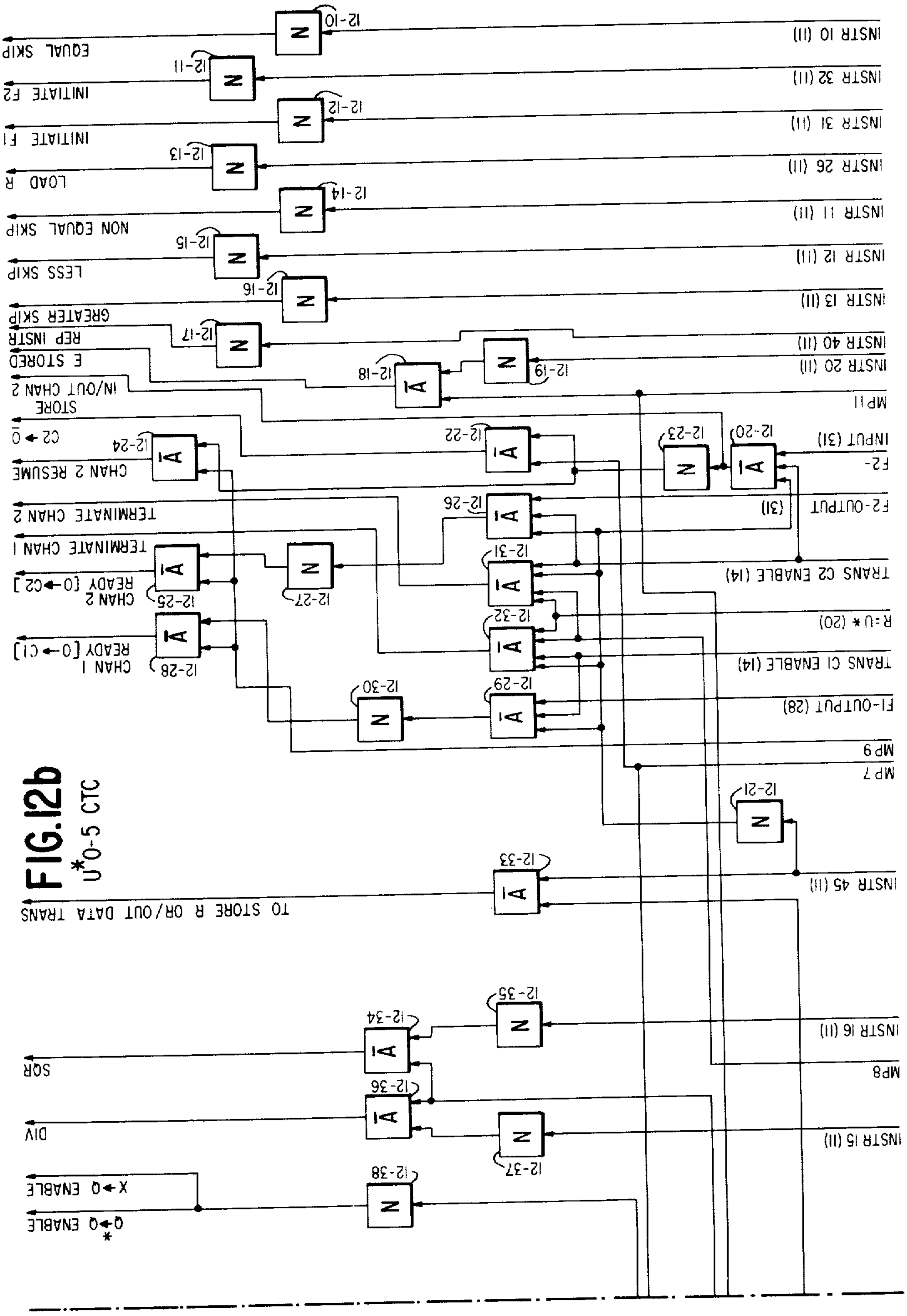


FIG. 12b
U*0-5 CTC

July 12, 1966

R. B. ARNDT ETAL

3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 24

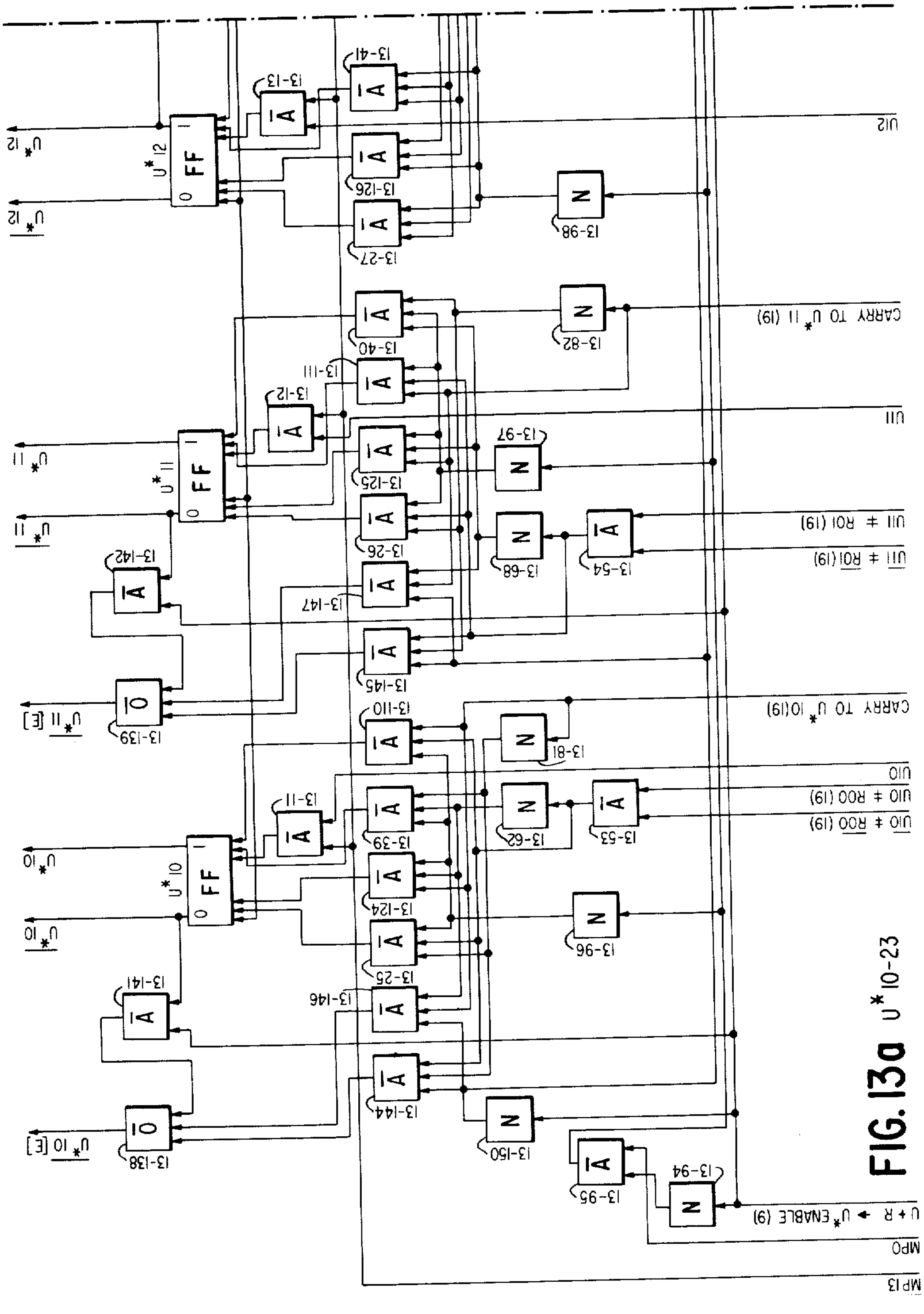


FIG. 13a U*10-23

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 25

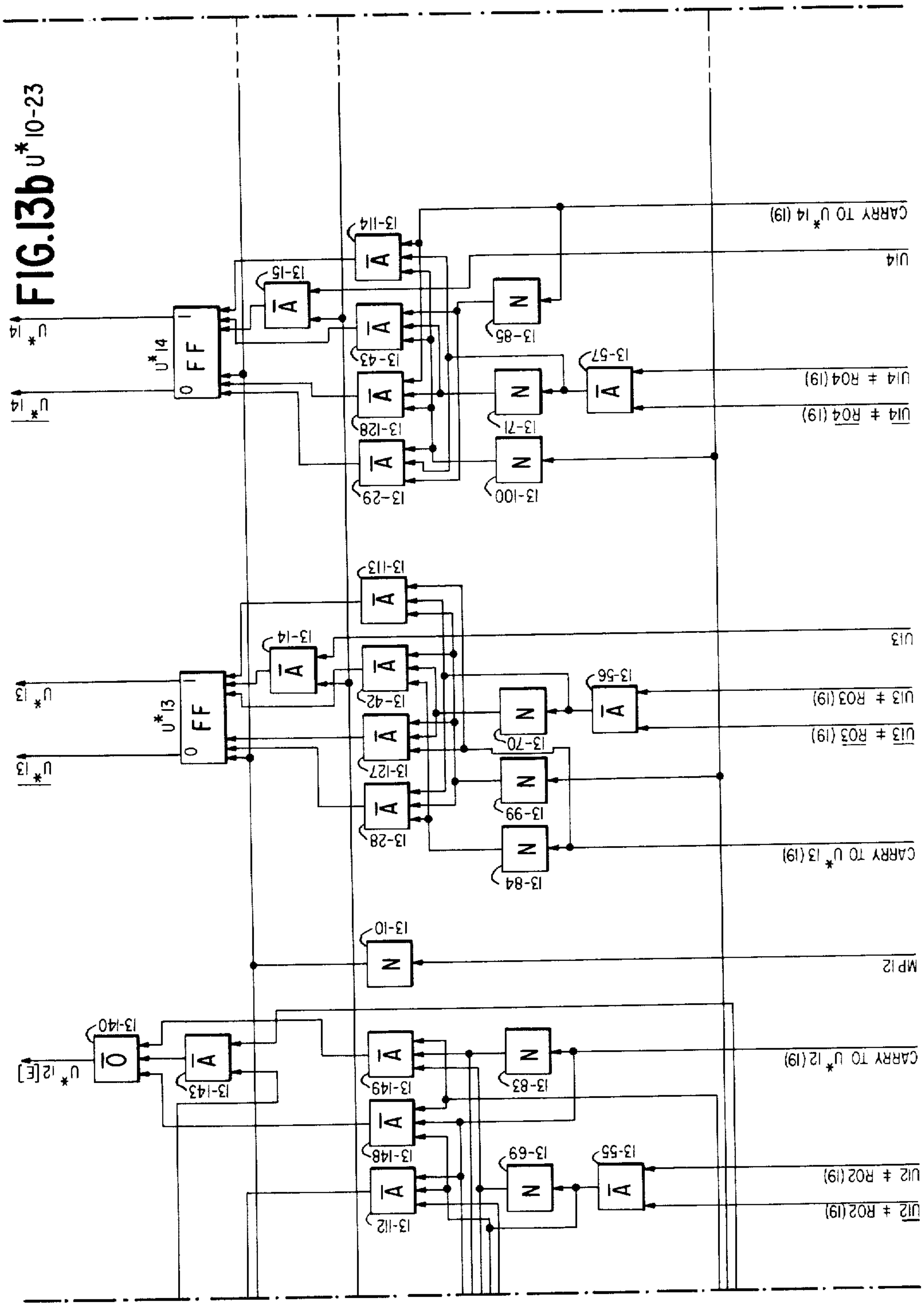


FIG.13b U^*_{10-23}

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 26

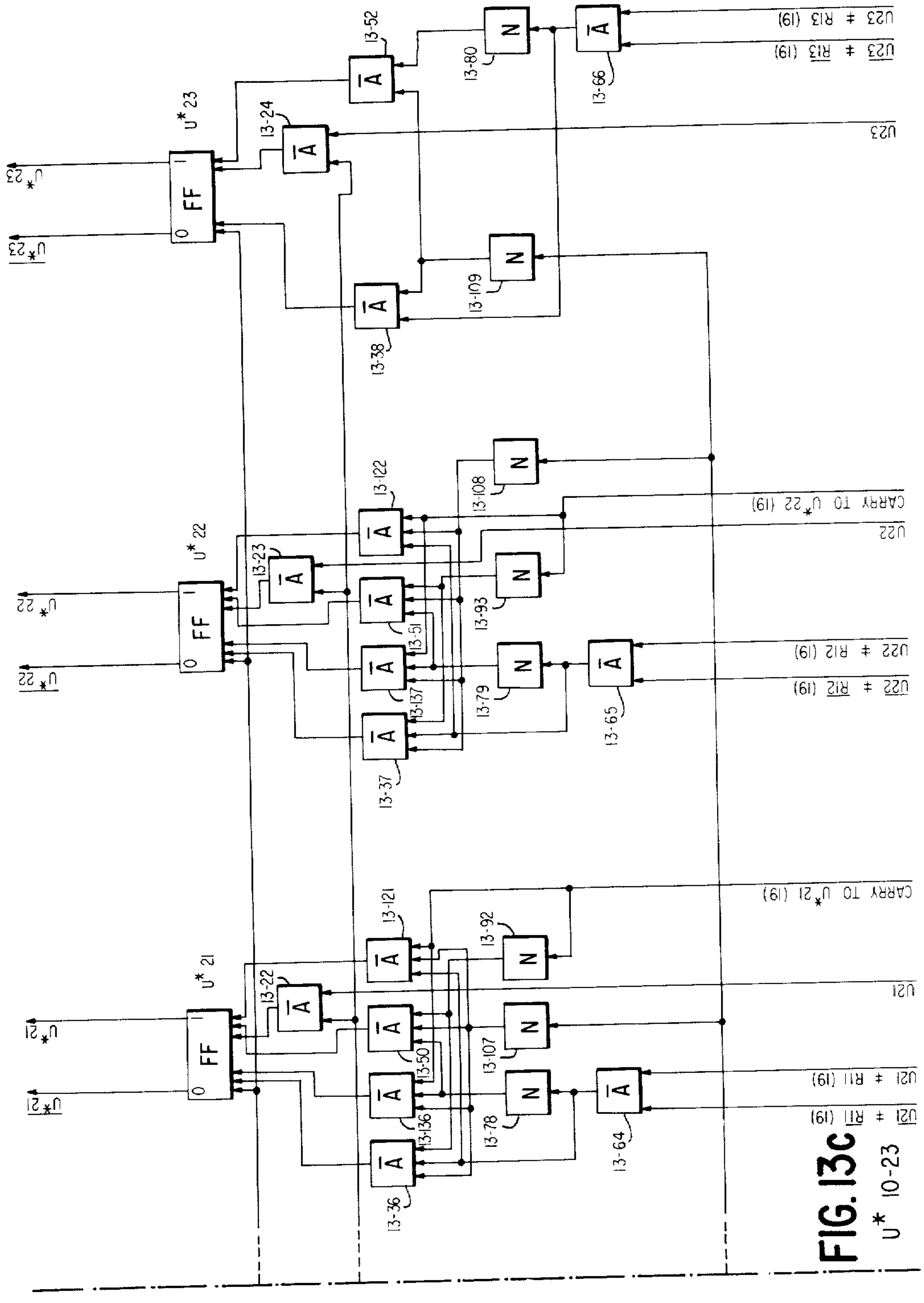


FIG. 13c
U* 10-23

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 27

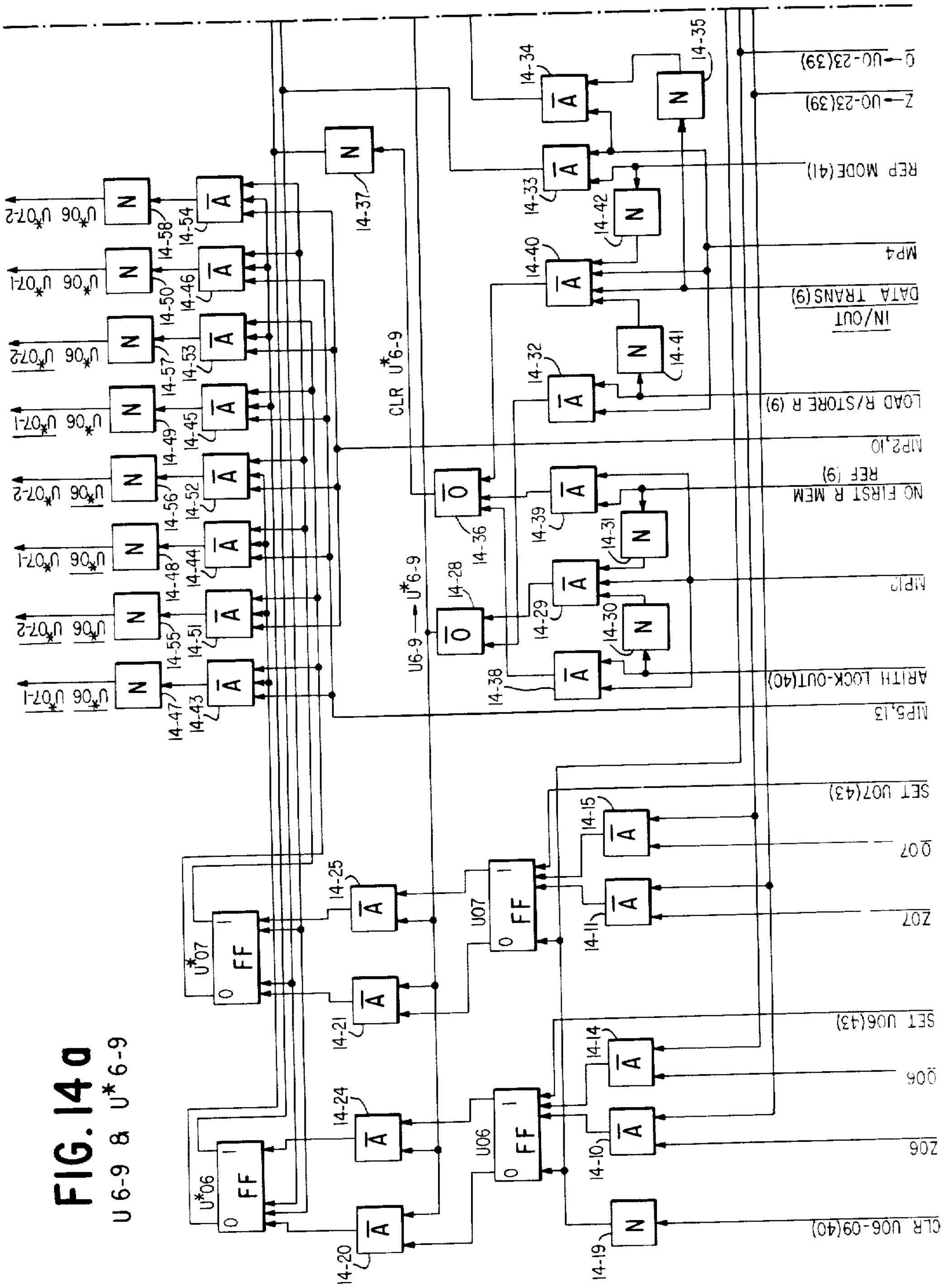


FIG. 14a
U 6-9 & U*6-9

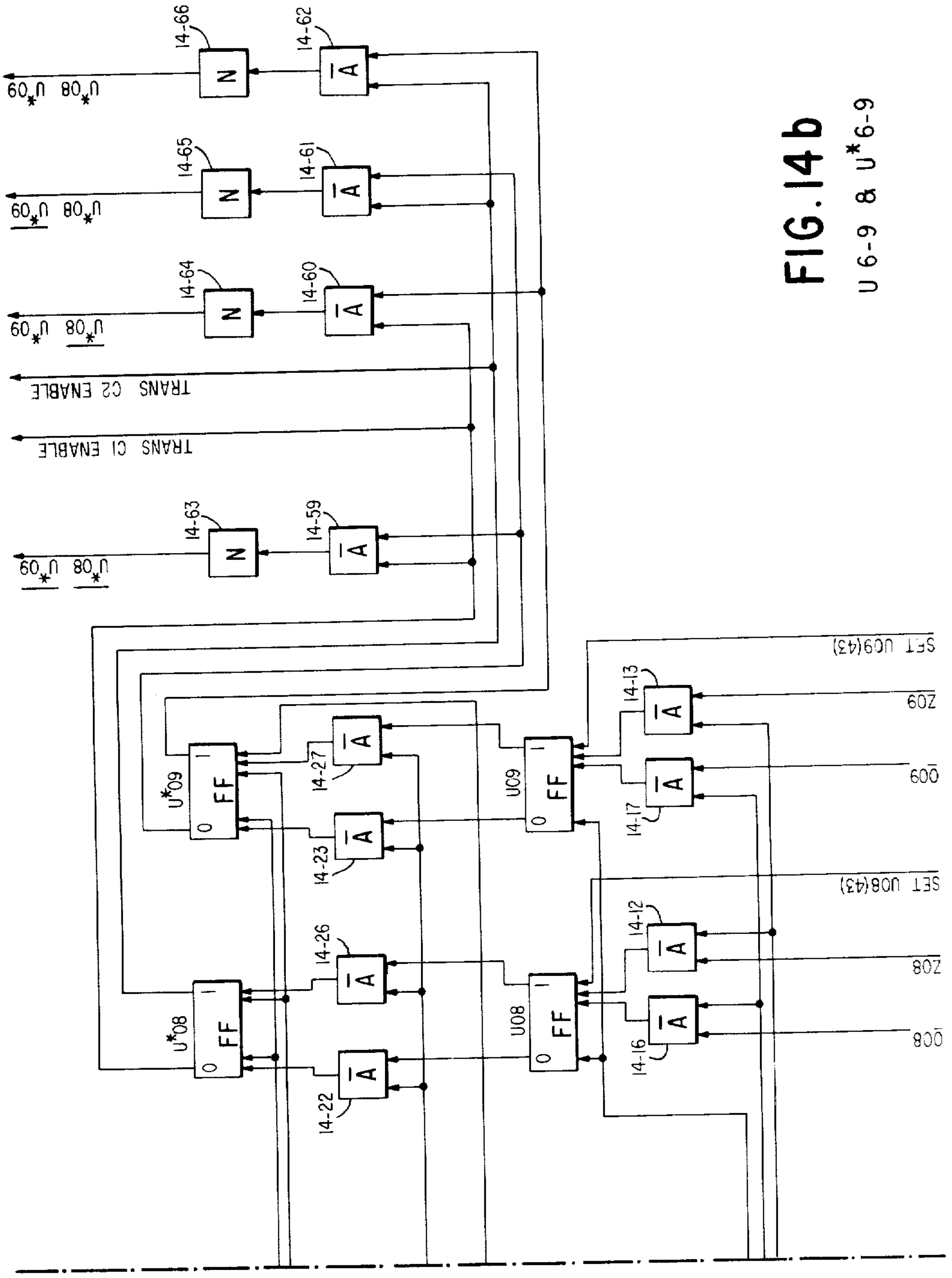


FIG. 14b

U 6-9 & U*6-9

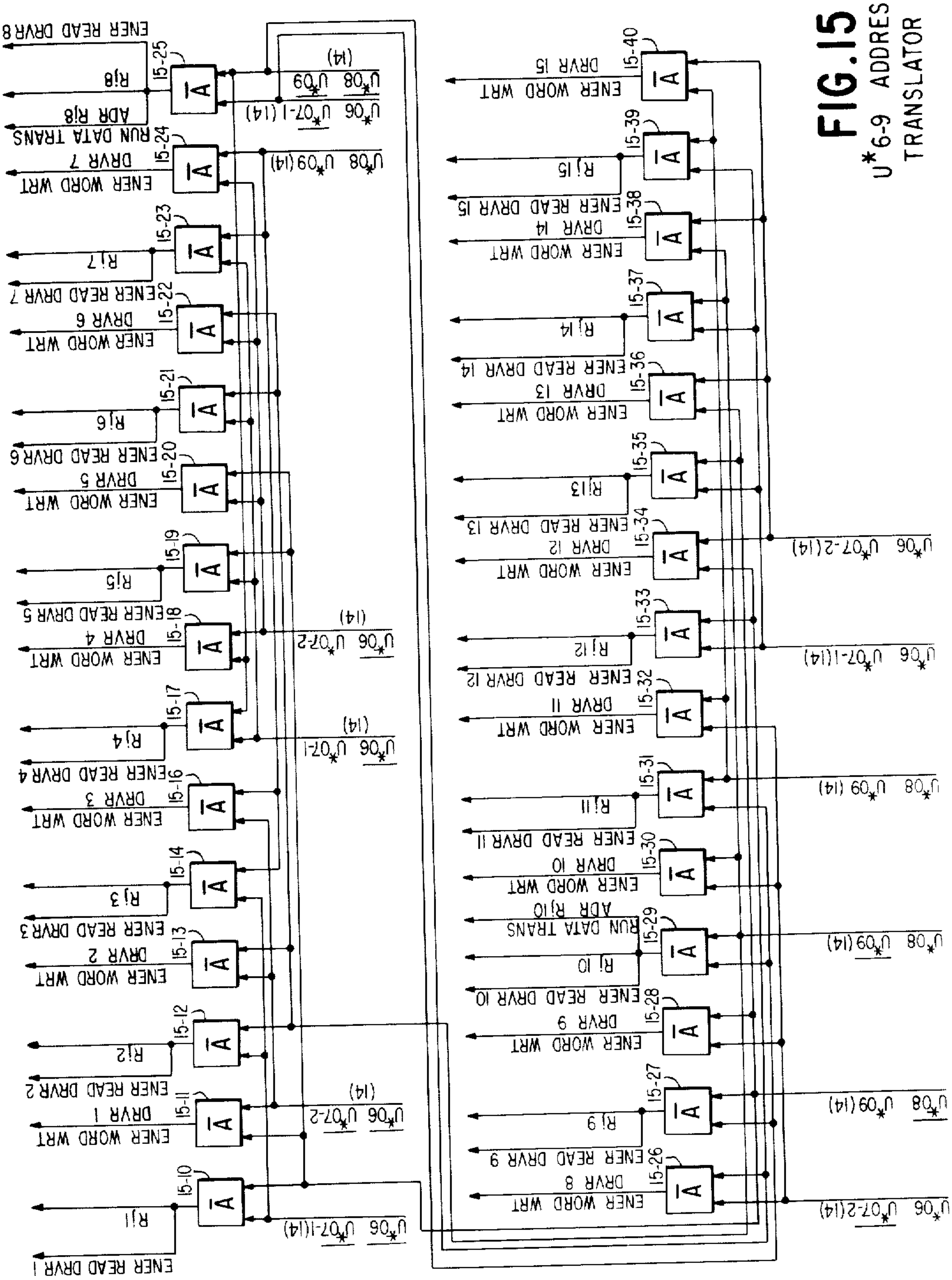


FIG. 15

U* 6-9 ADDRESS TRANSLATOR

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 30

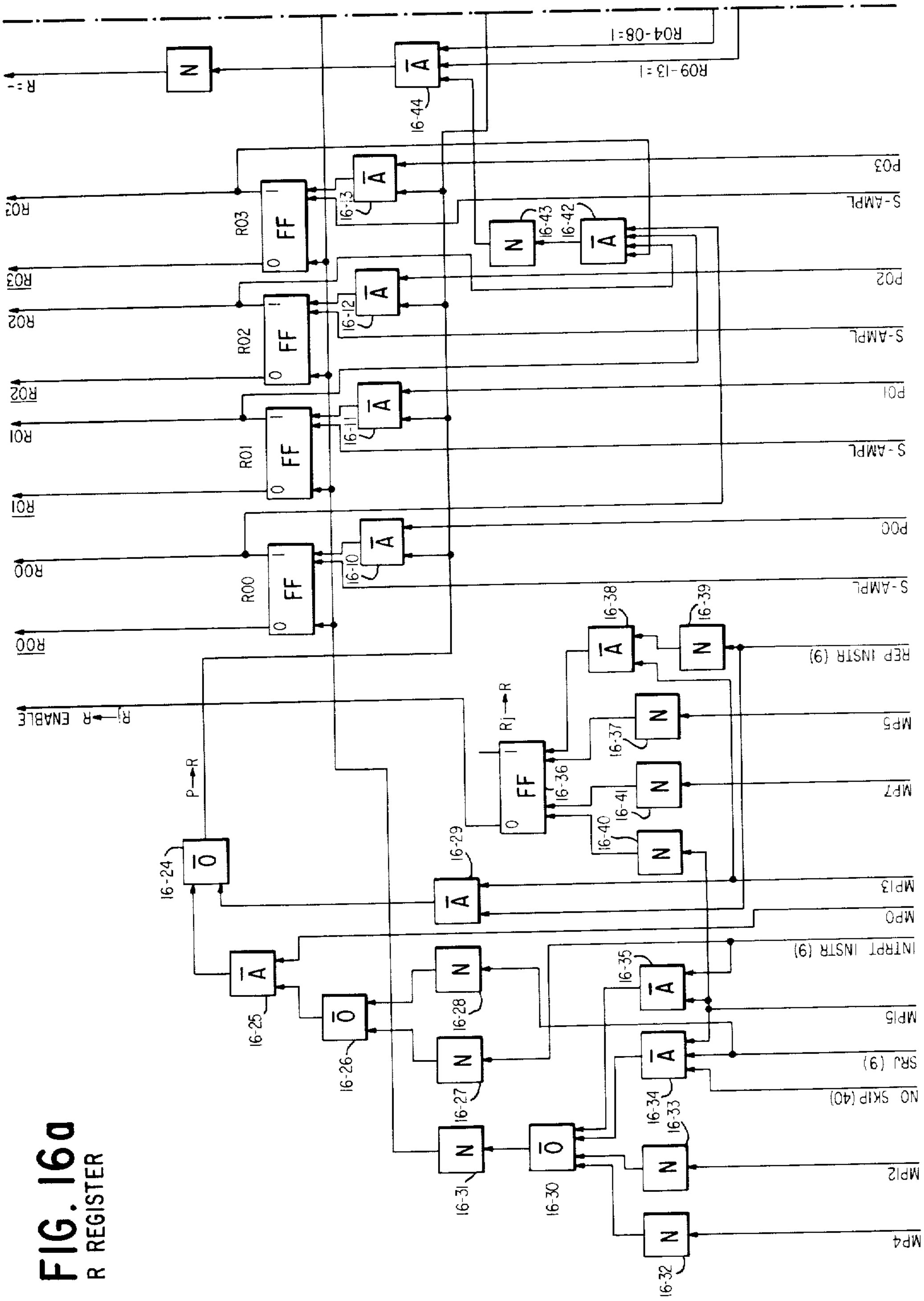


FIG. 16a
R REGISTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 31

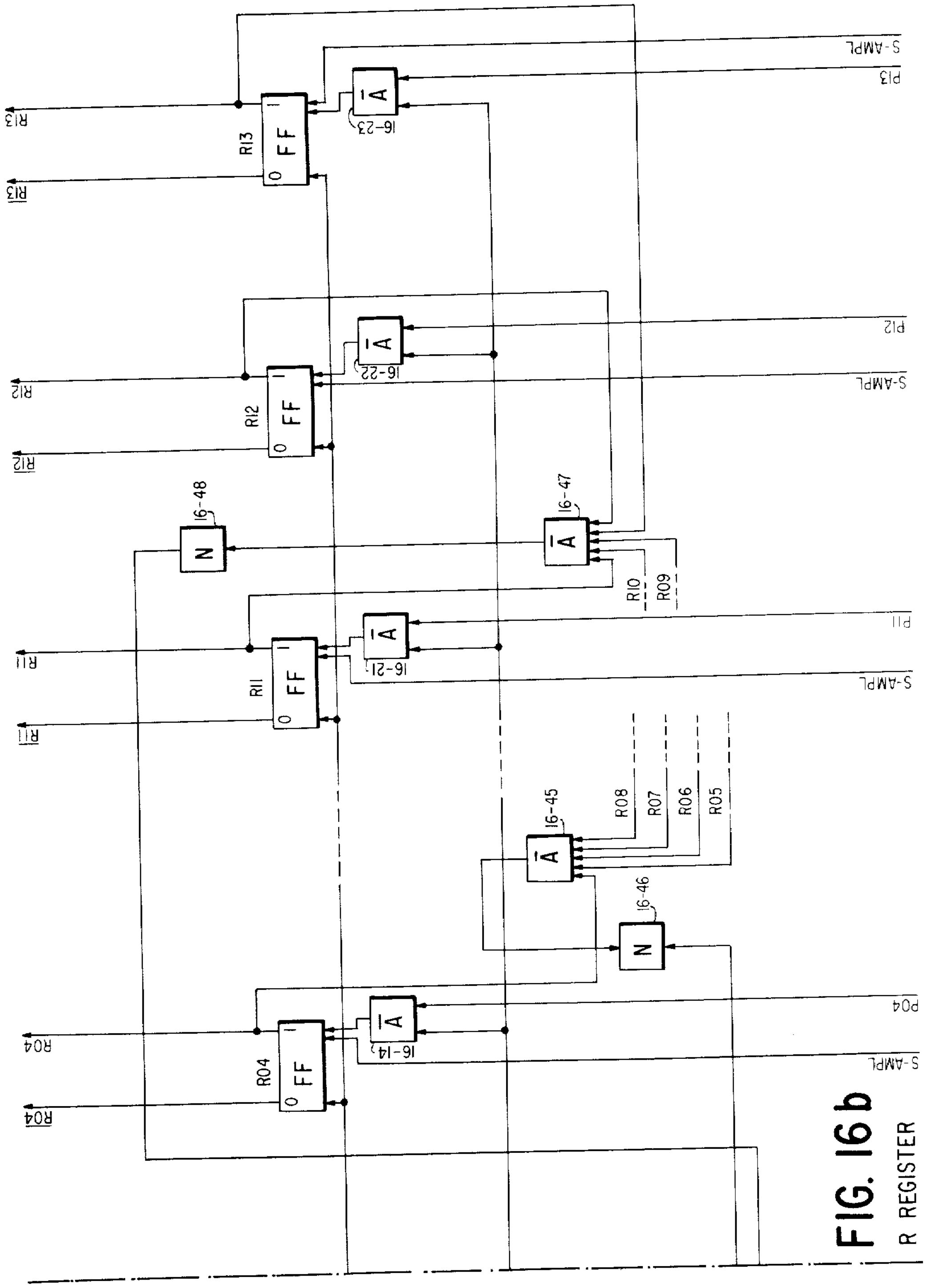


FIG. 16b

R REGISTER

July 12, 1966 R. B. ARNDT ET AL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 32

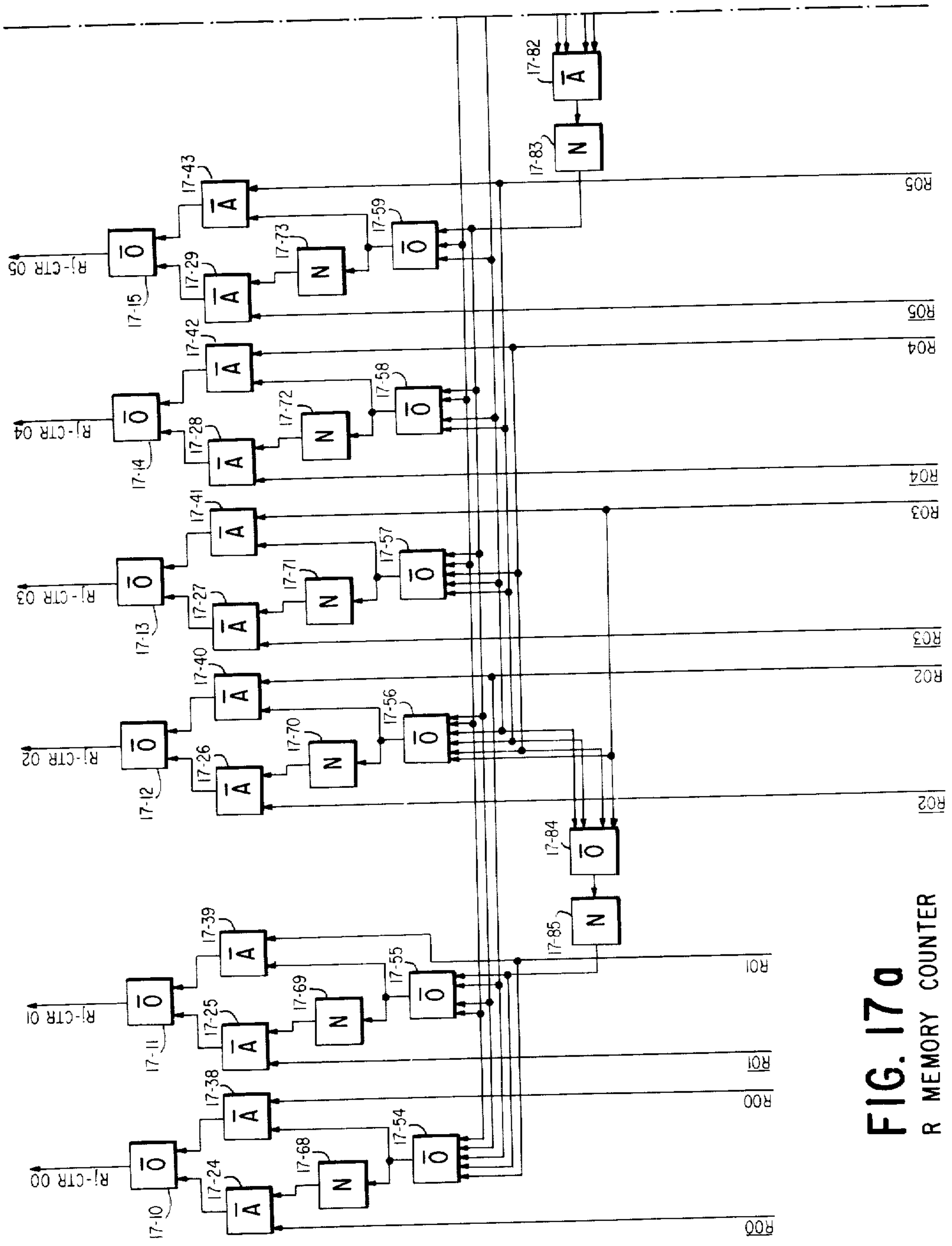


FIG. 17a
R MEMORY COUNTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 33

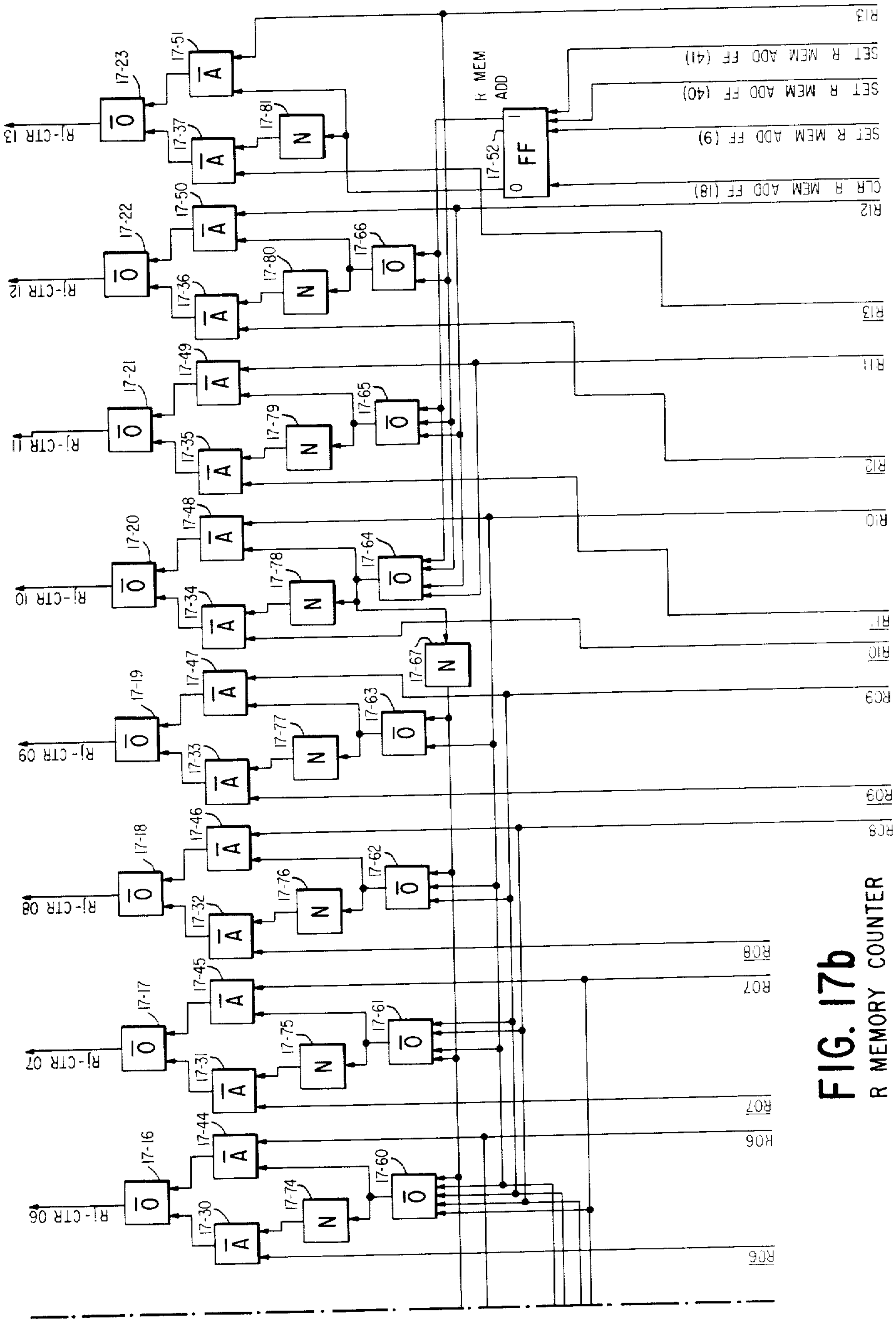


FIG. 17b

R MEMORY COUNTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 34

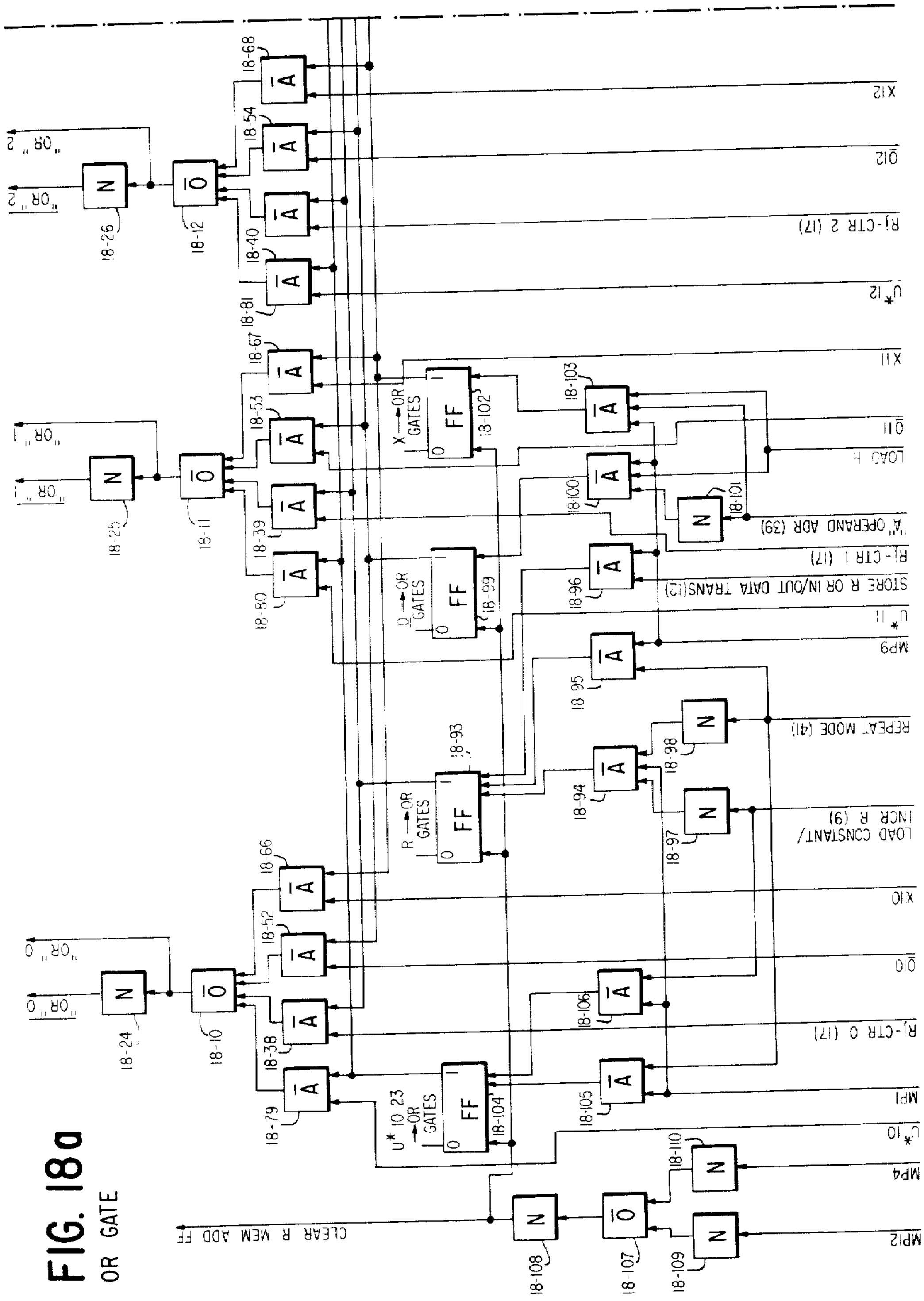


FIG. 18a

OR GATE

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 35

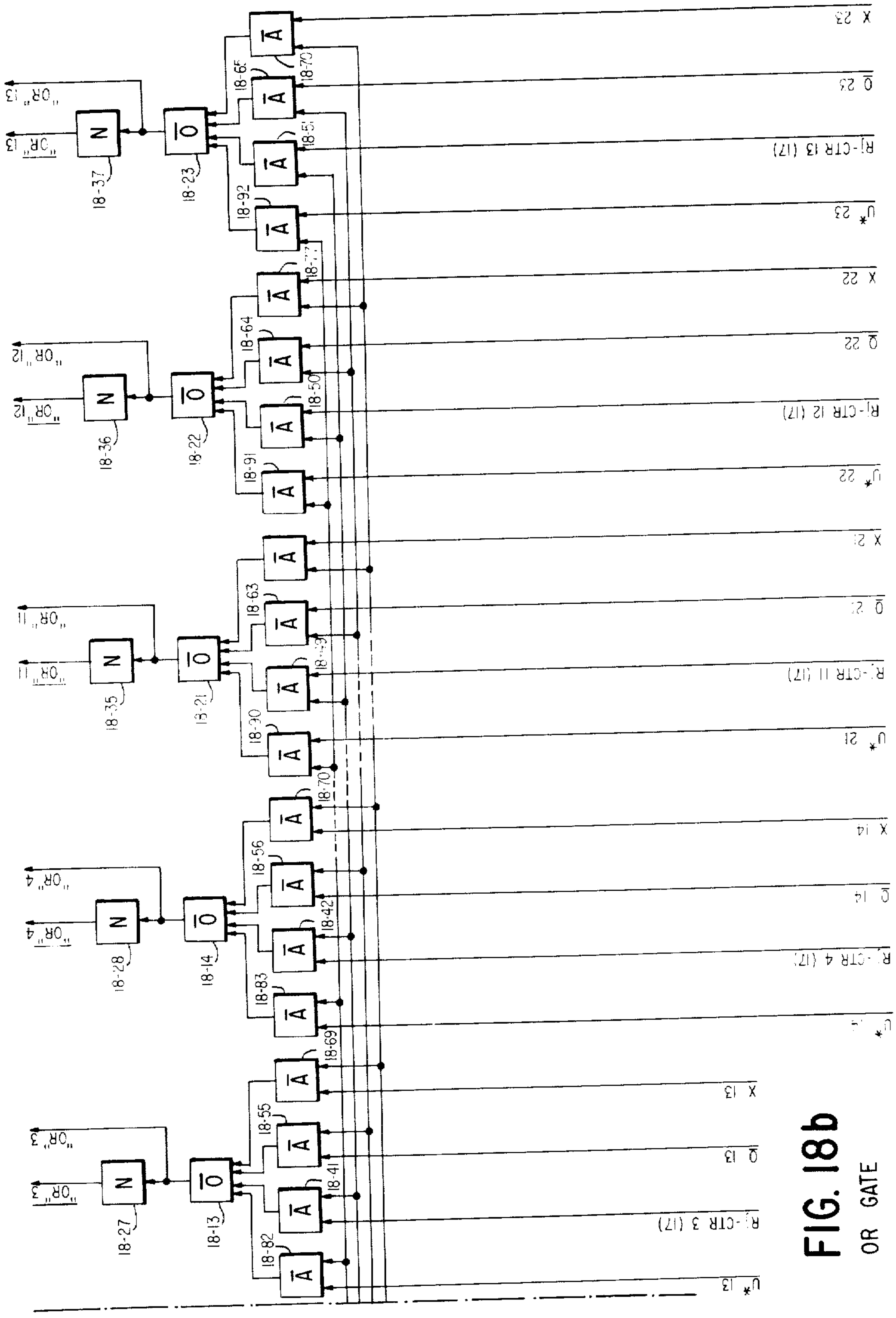


FIG. 18b

OR GATE

STORED PROGRAM SYSTEM

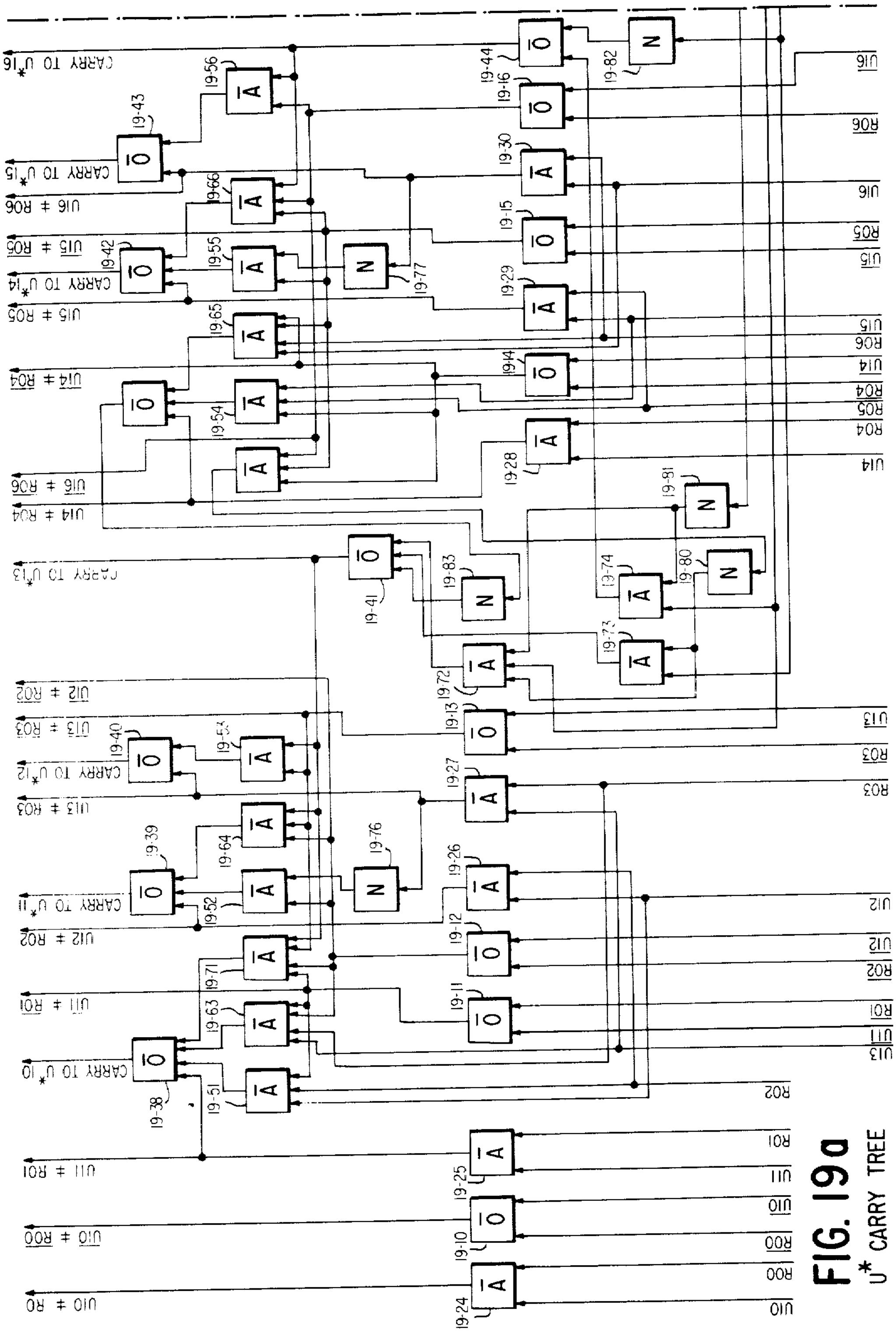


FIG. 19a

U* CARRY TREE

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

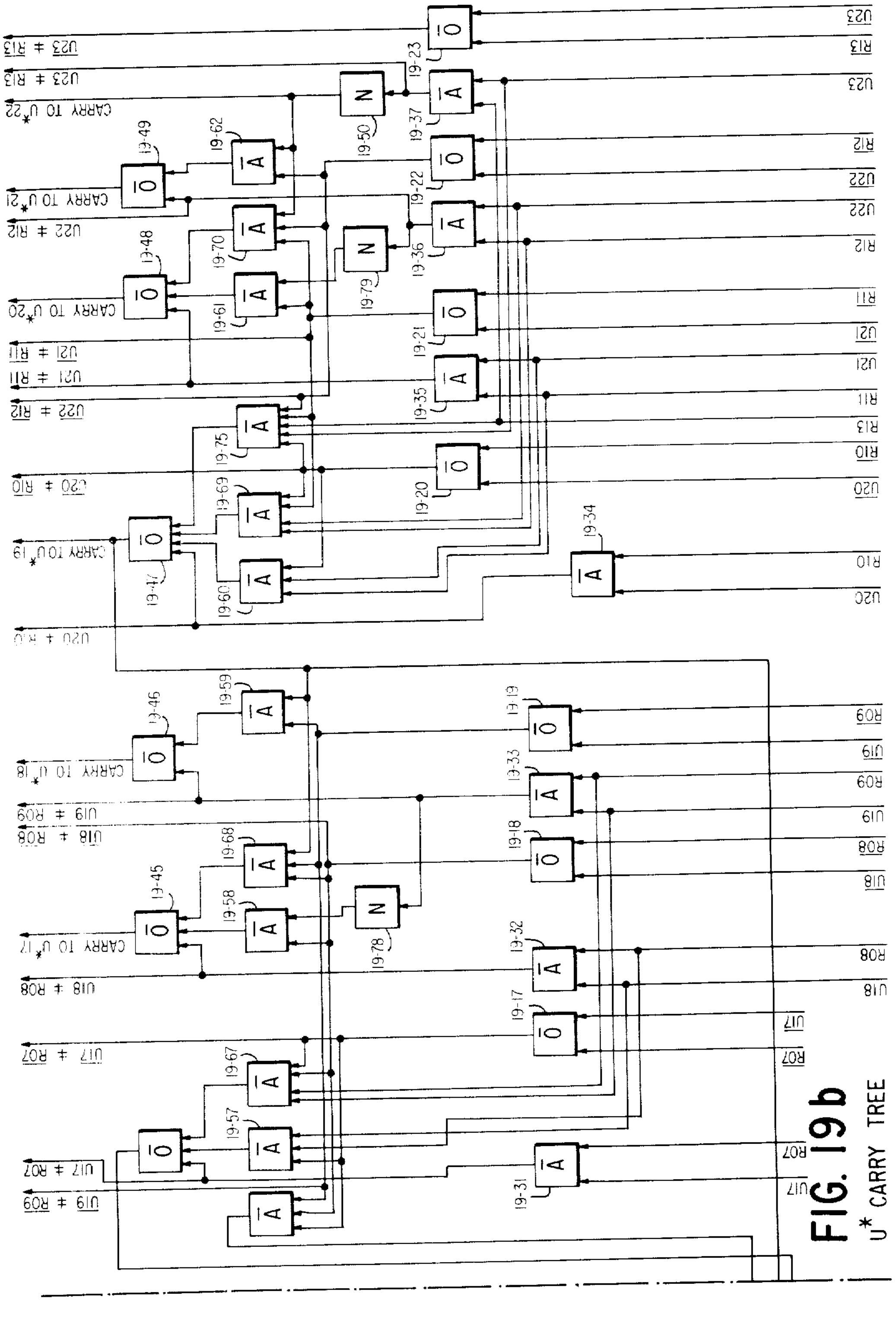


FIG. 19b

U* CARRY TREE

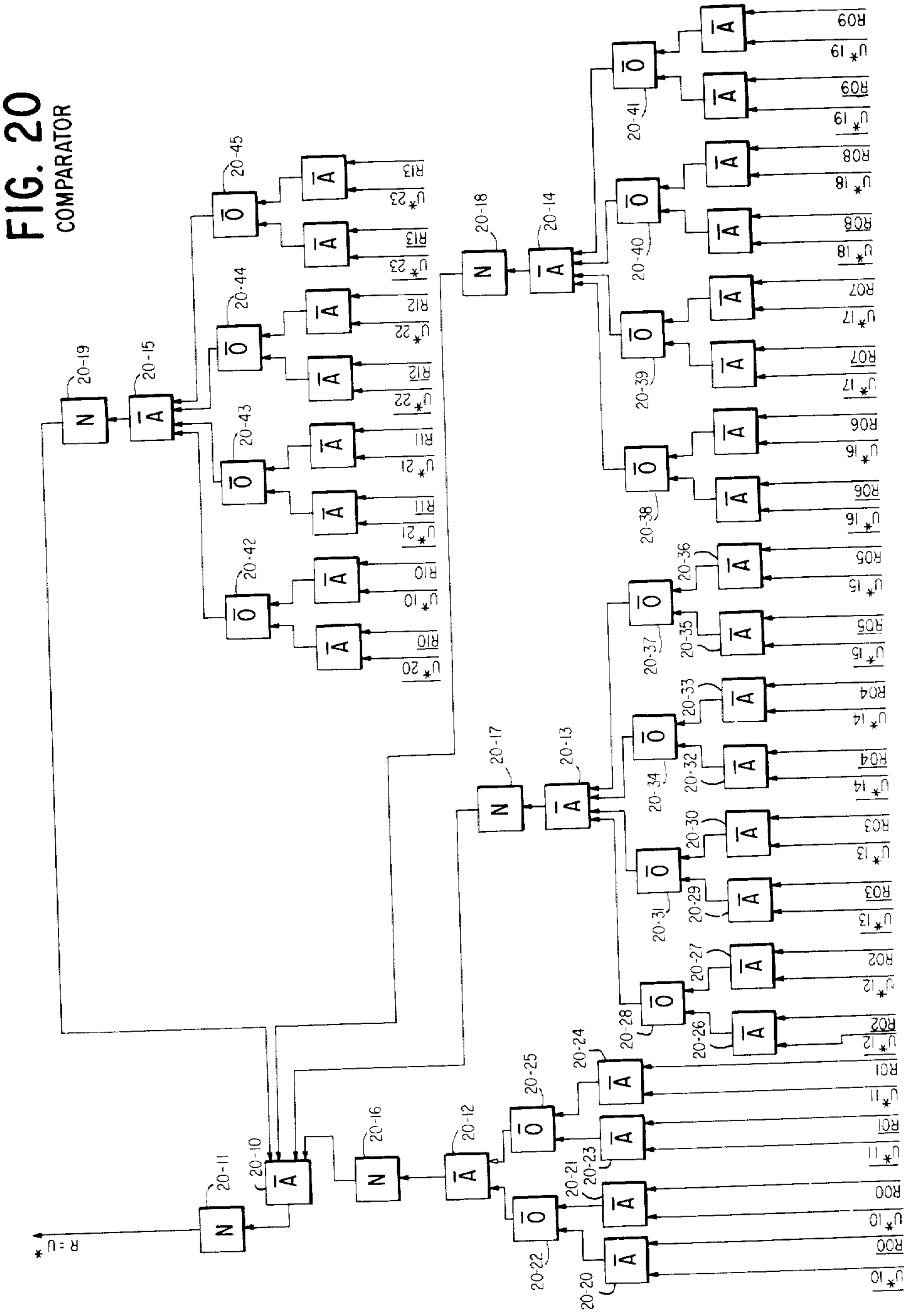
July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 38

FIG. 20

COMPARATOR



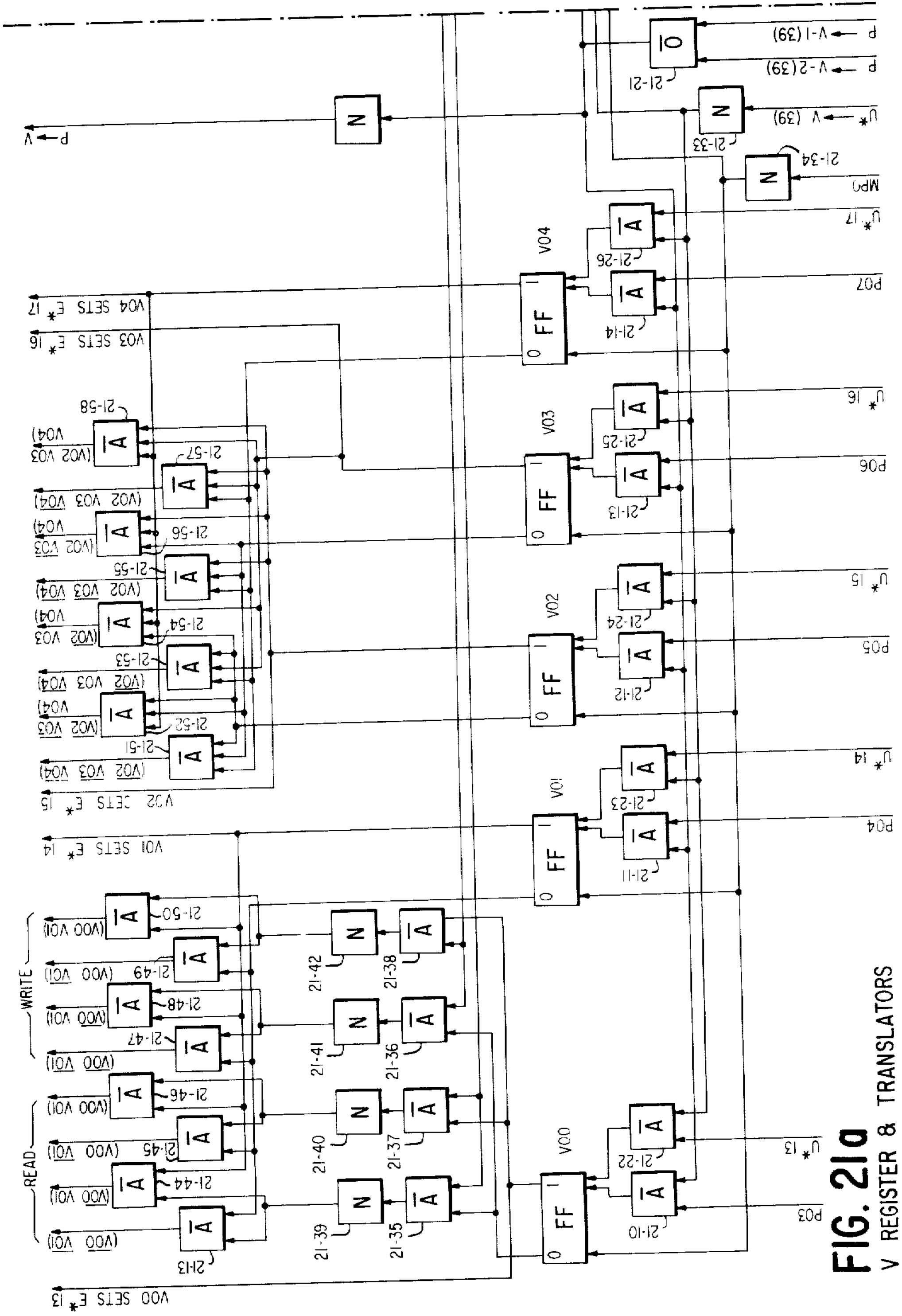


FIG. 21a
V REGISTER & TRANSLATORS

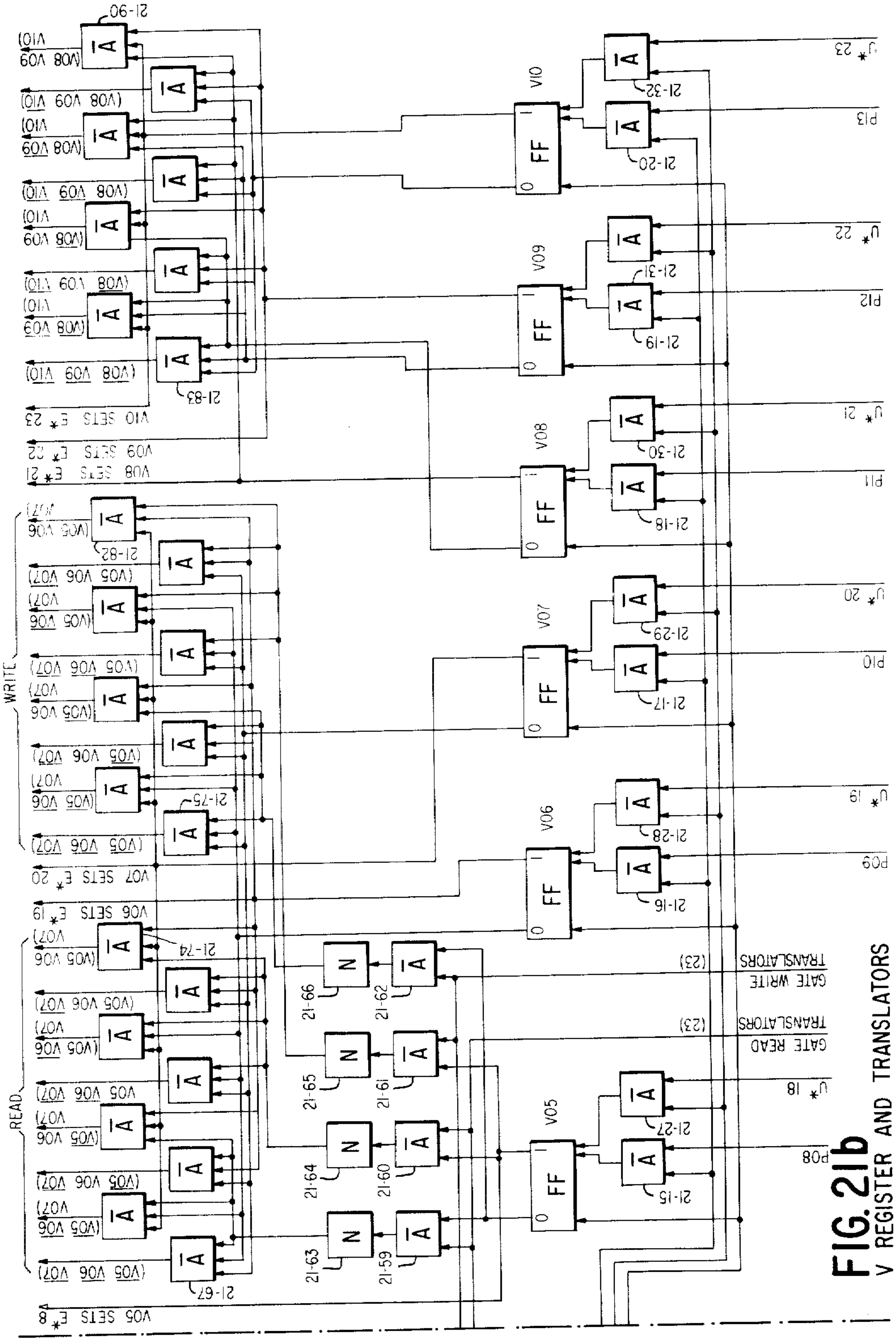


FIG. 21b
V REGISTER AND TRANSLATORS

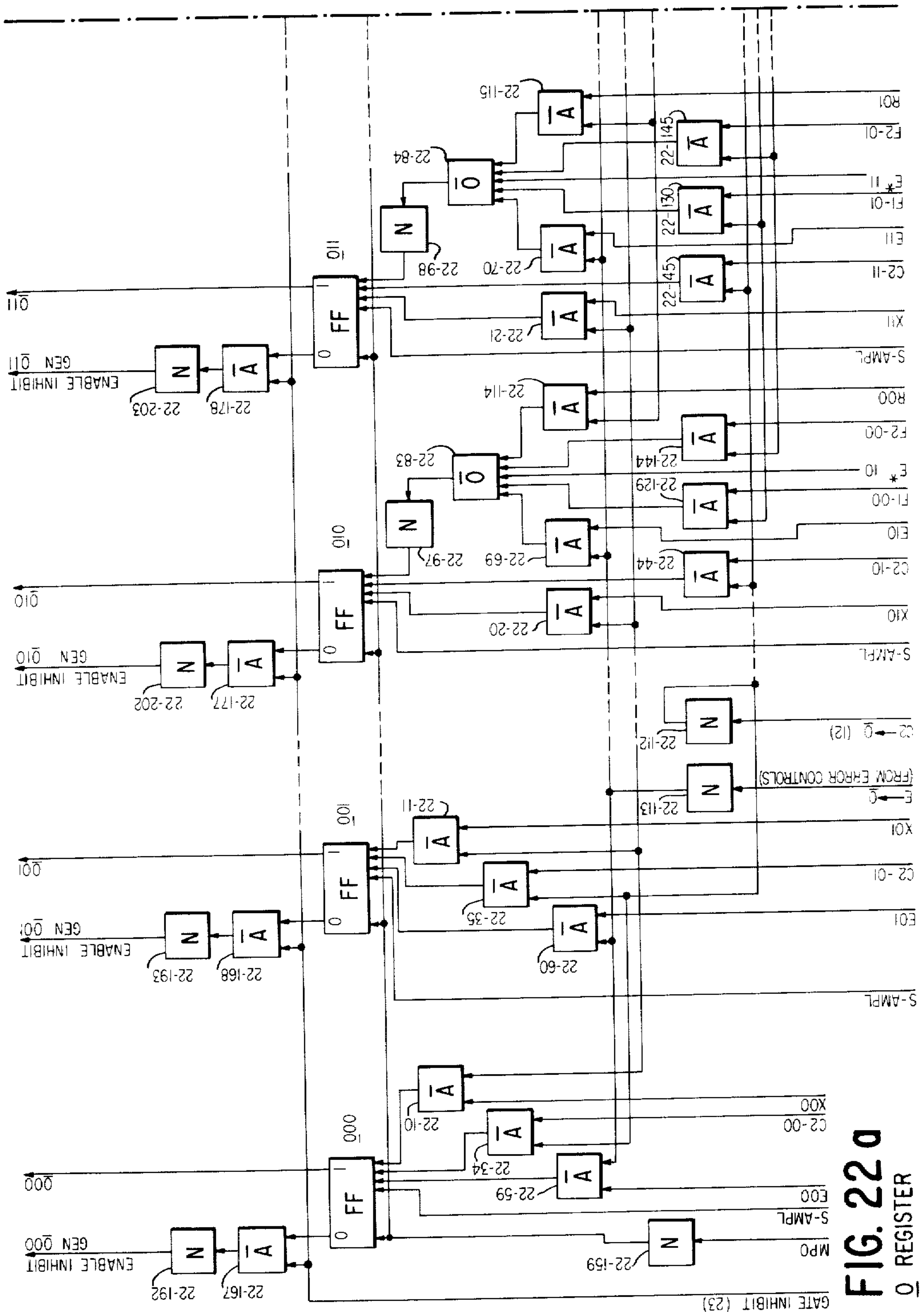


FIG. 22a

0 REGISTER

STORED PROGRAM SYSTEM

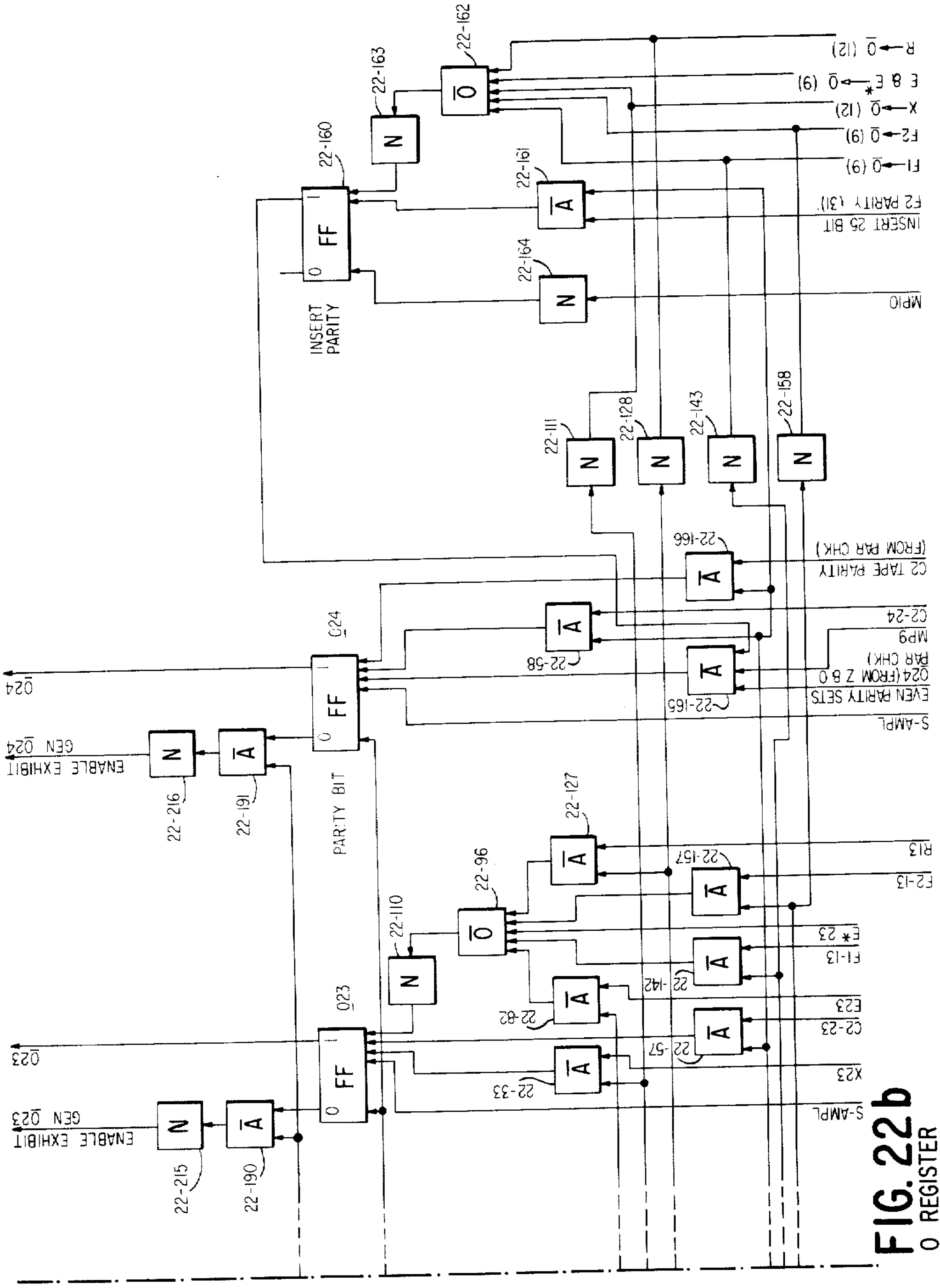


FIG. 22b
Q REGISTER

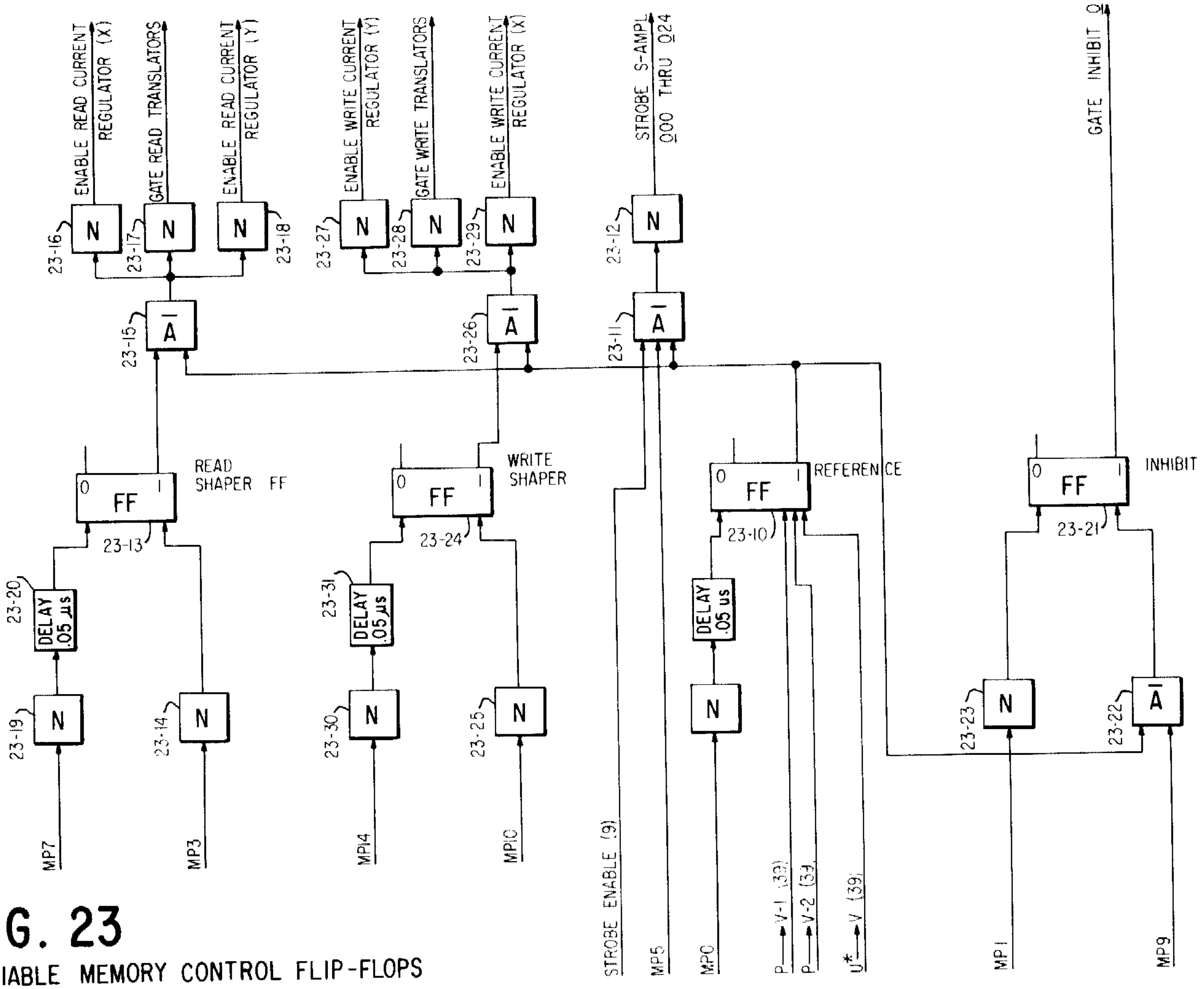


FIG. 23
VARIABLE MEMORY CONTROL FLIP-FLOPS

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 44

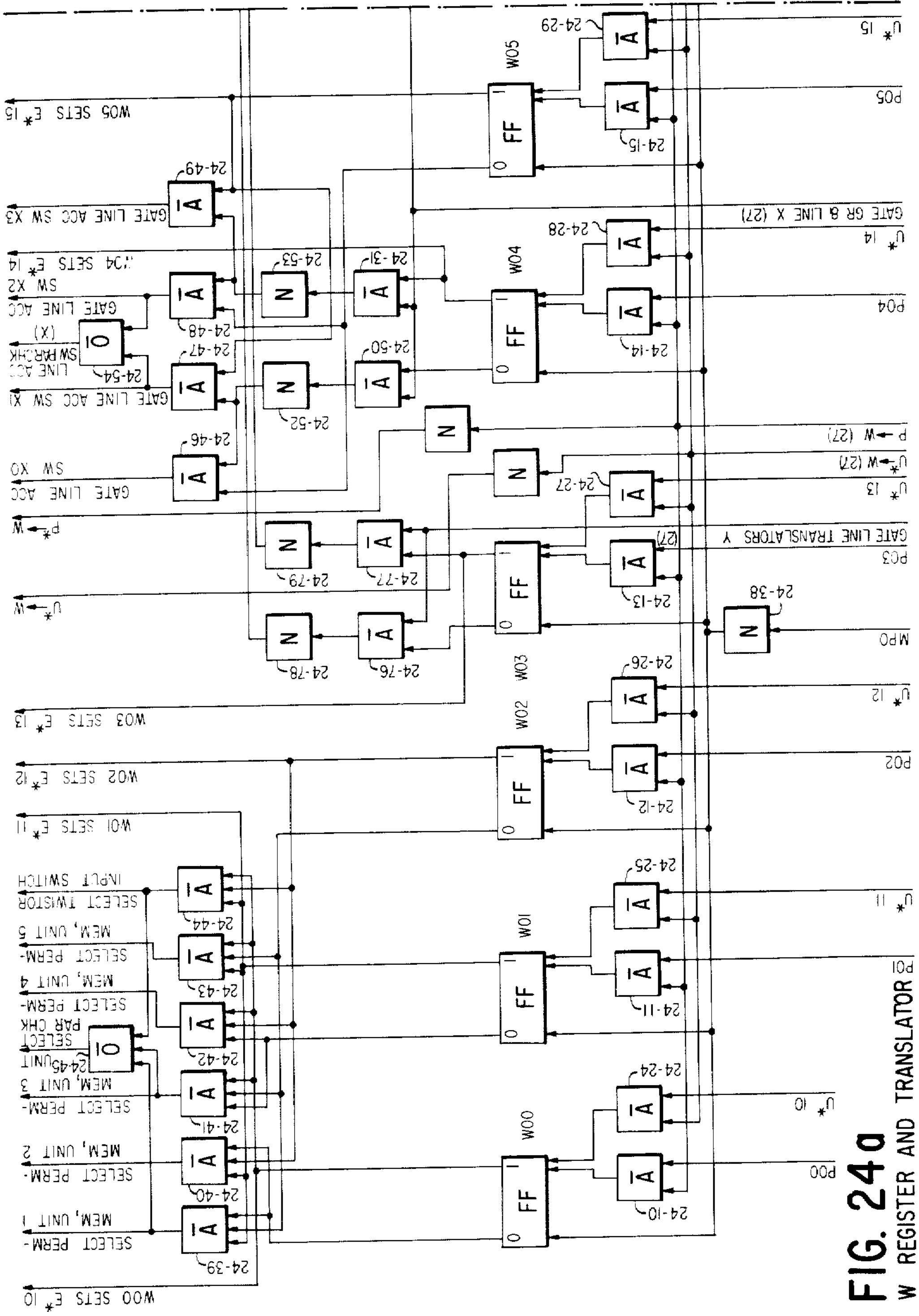


FIG. 24a
W REGISTER AND TRANSLATOR

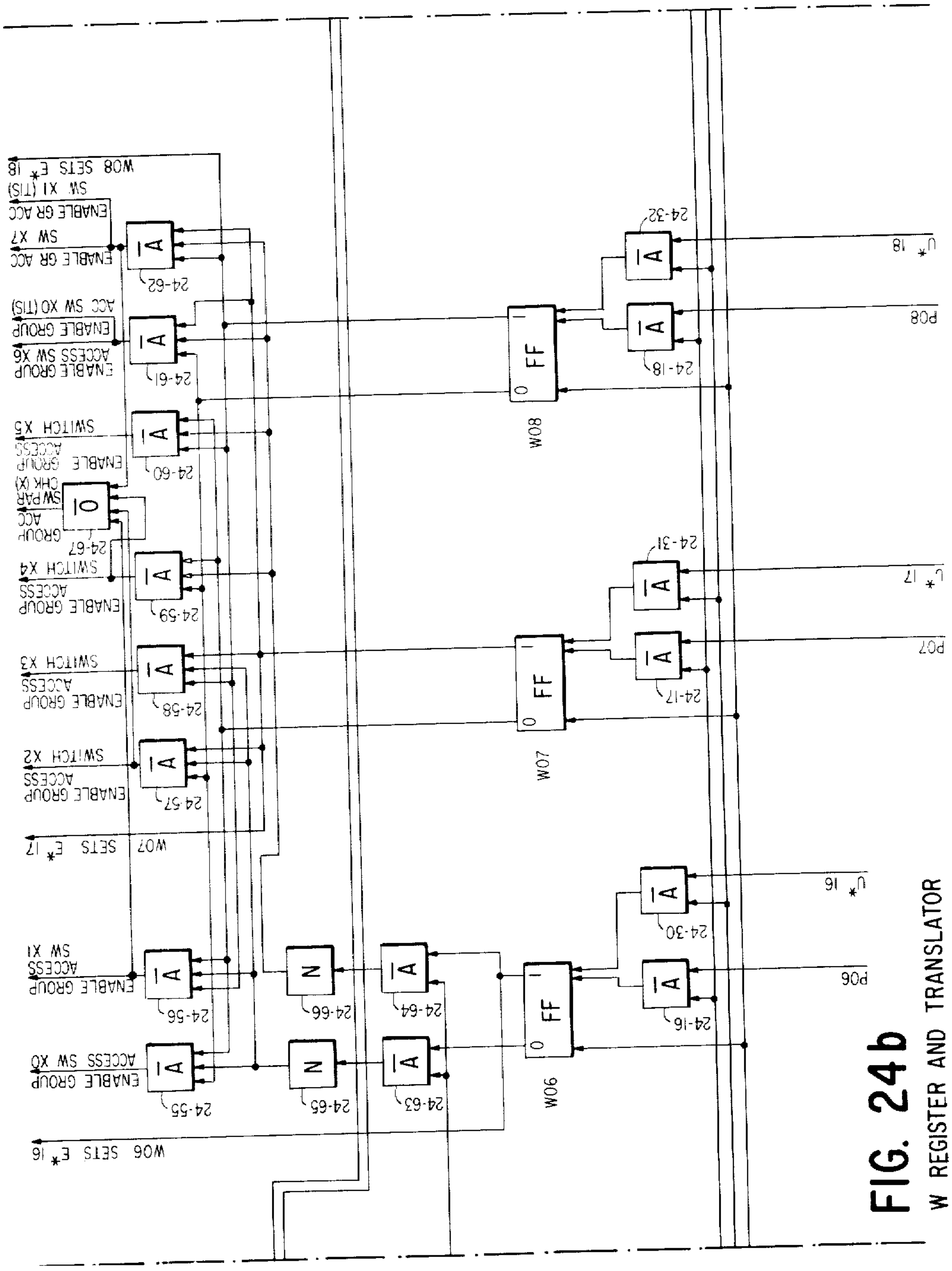


FIG. 24b

W REGISTER AND TRANSLATOR

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 46

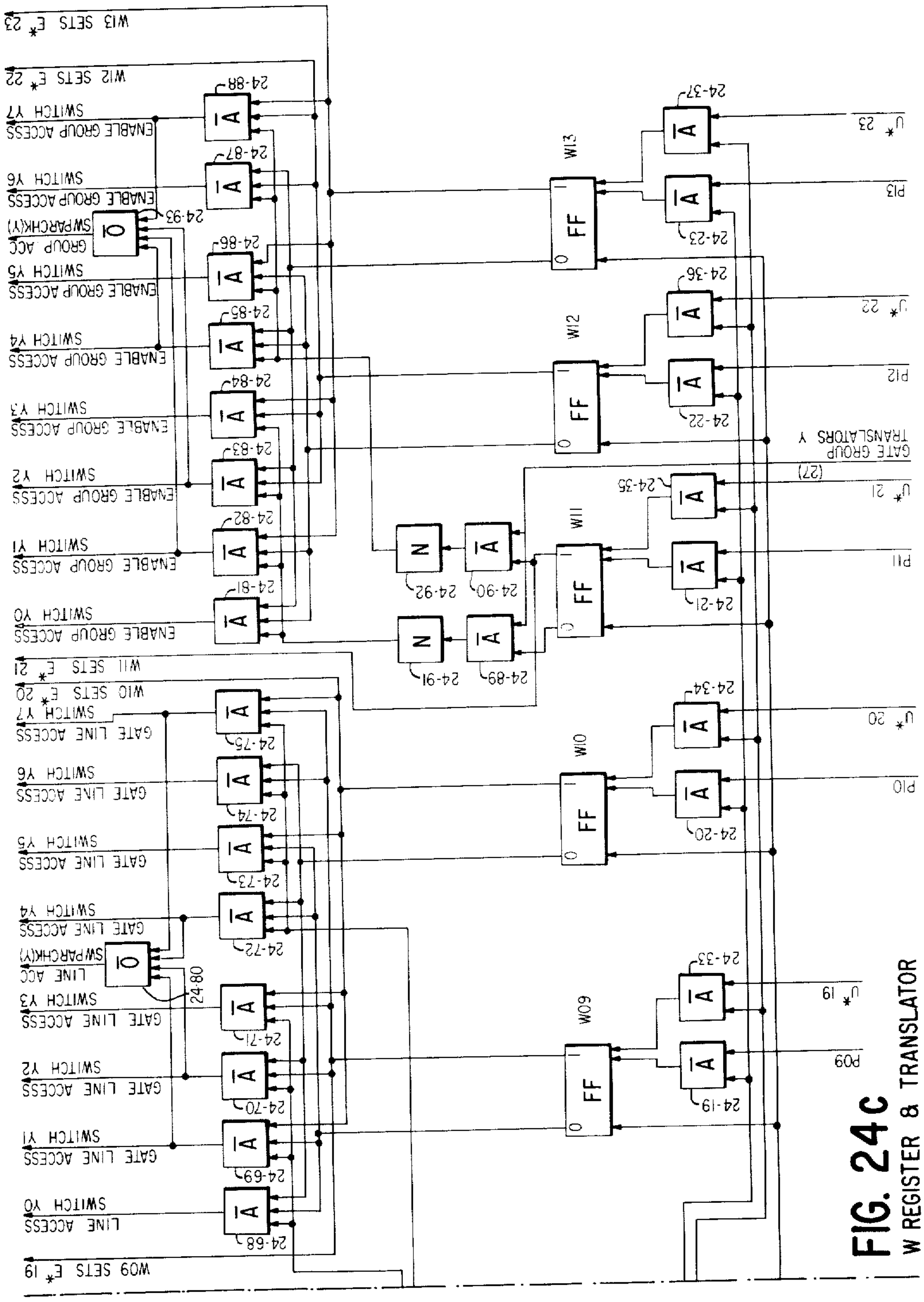


FIG. 24c
W REGISTER & TRANSLATOR

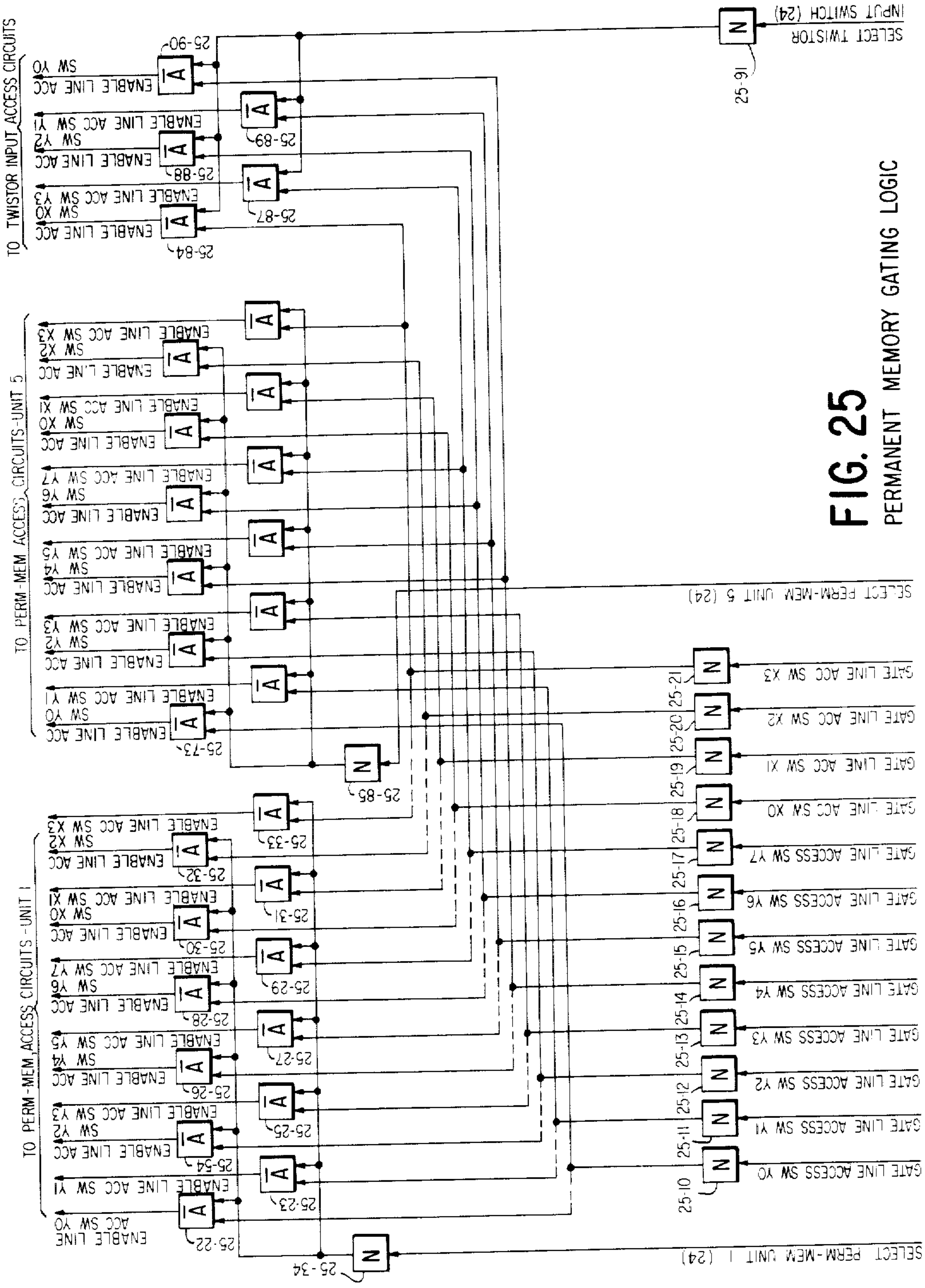


FIG. 25
PERMANENT MEMORY GATING LOGIC

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 48

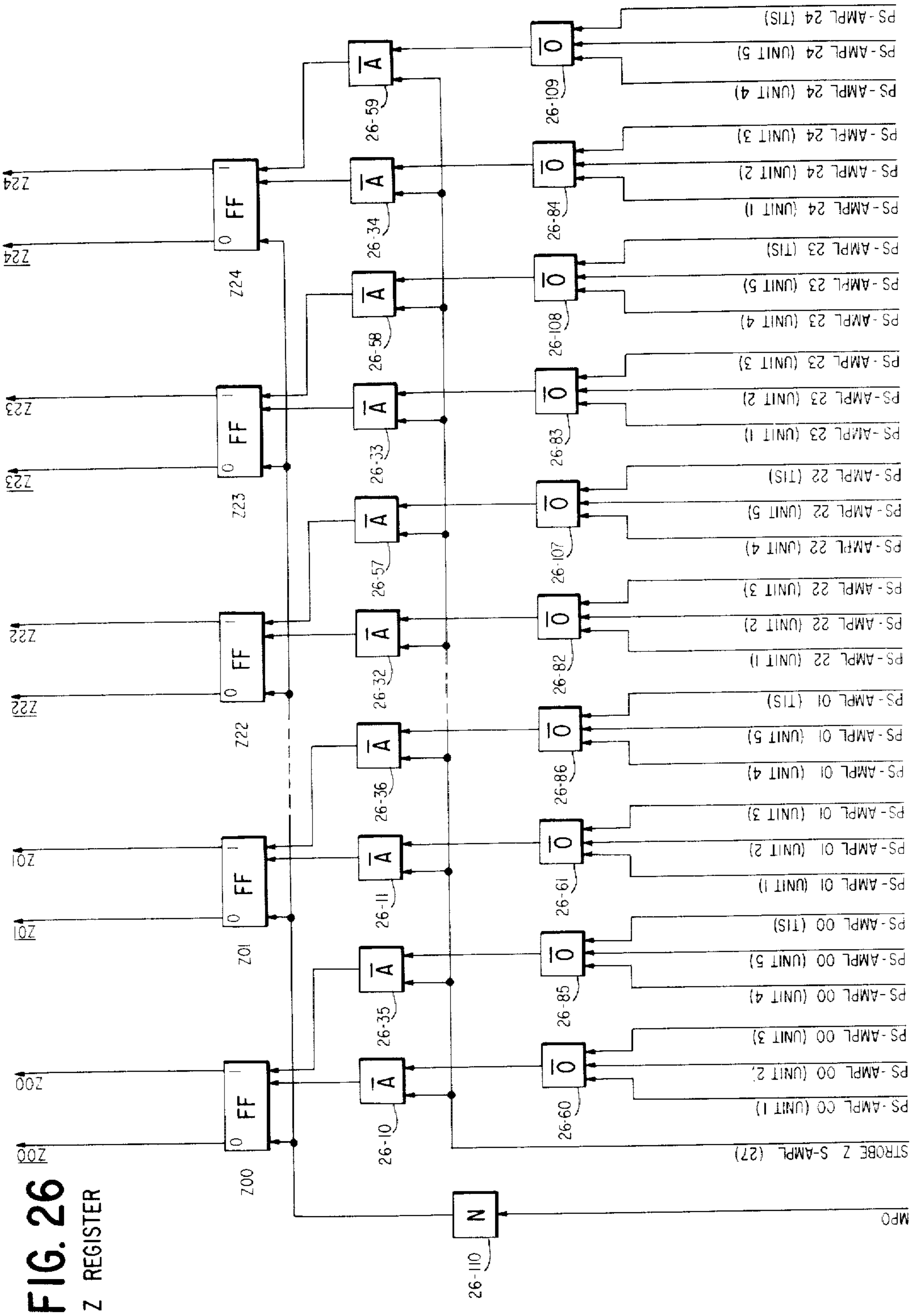


FIG. 26
Z REGISTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 49

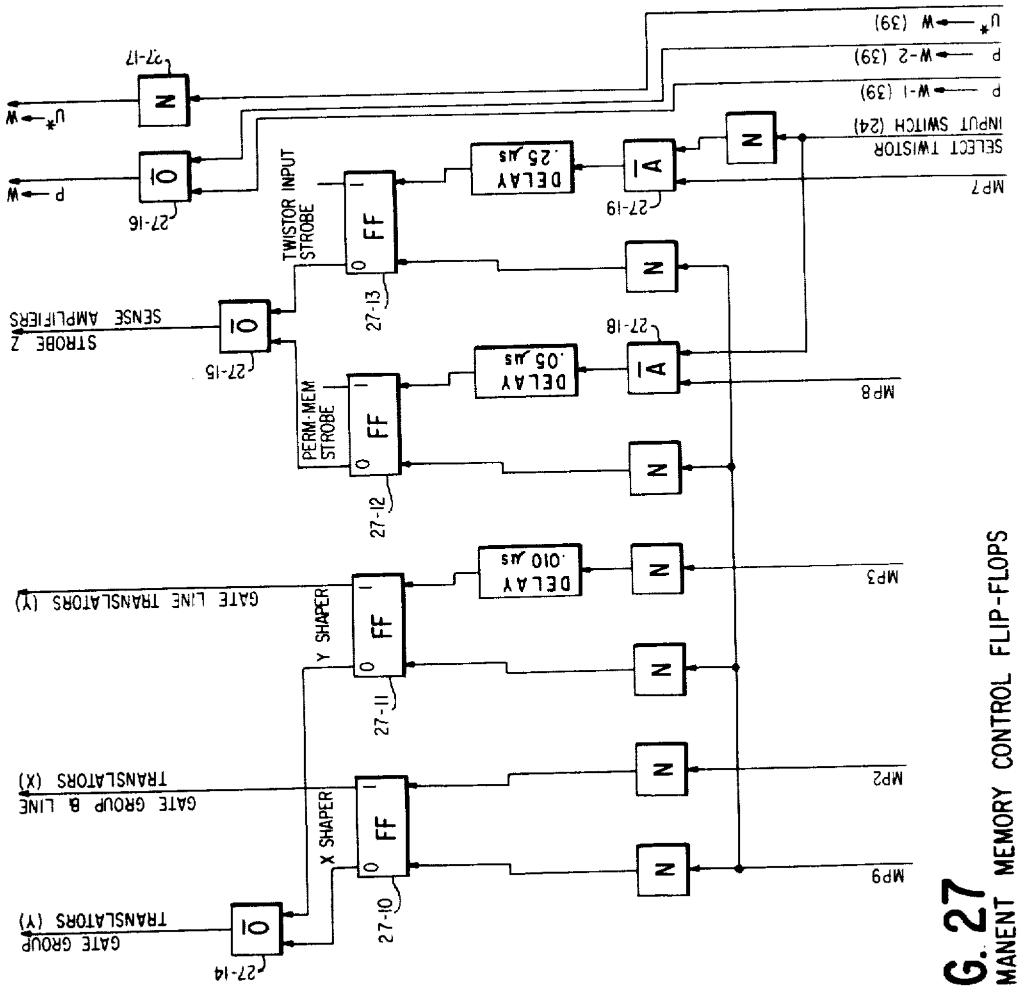


FIG. 27 PERMANENT MEMORY CONTROL FLIP-FLOPS

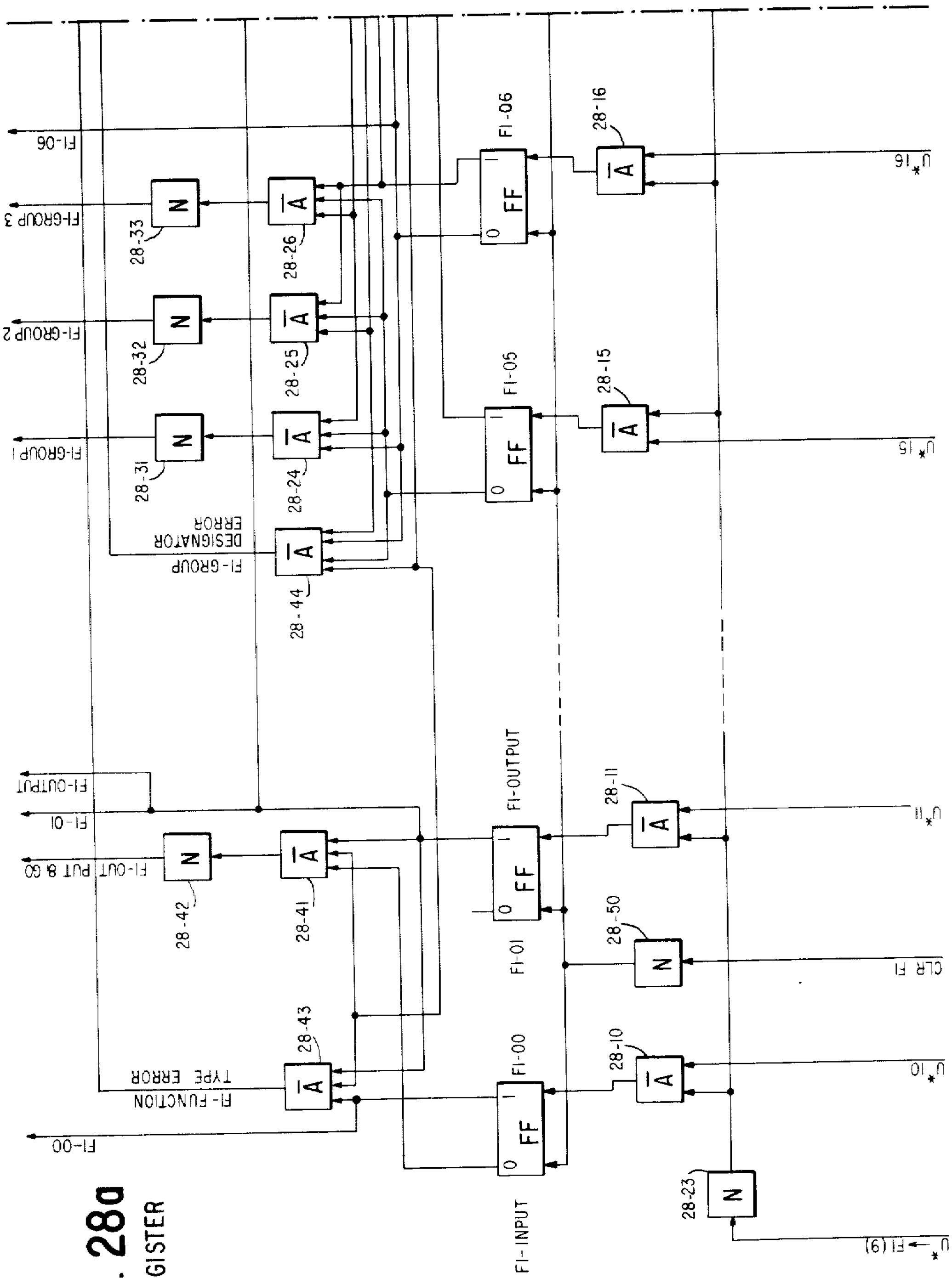
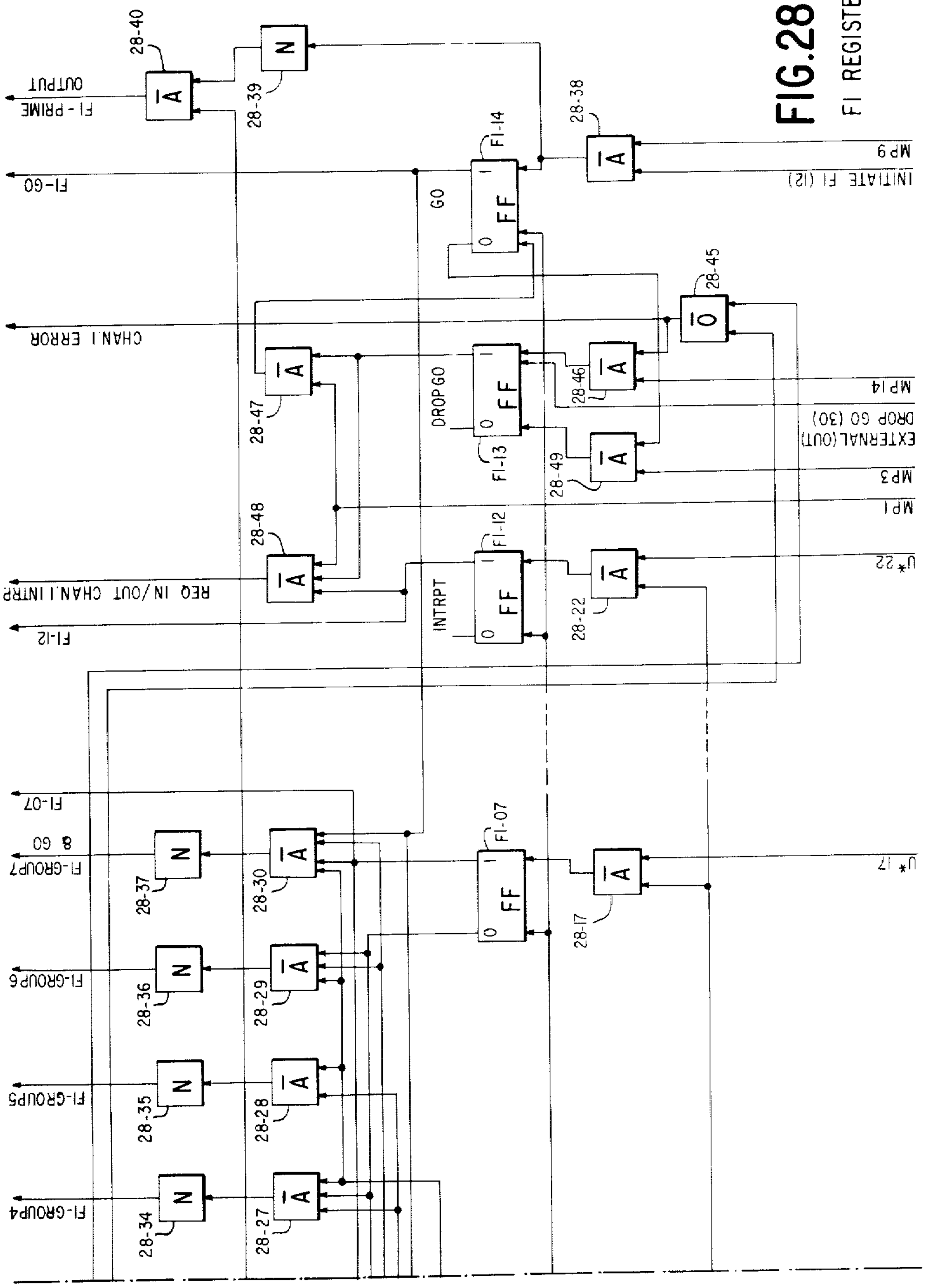


FIG. 28a
FI REGISTER

FIG.28b

FI REGISTER



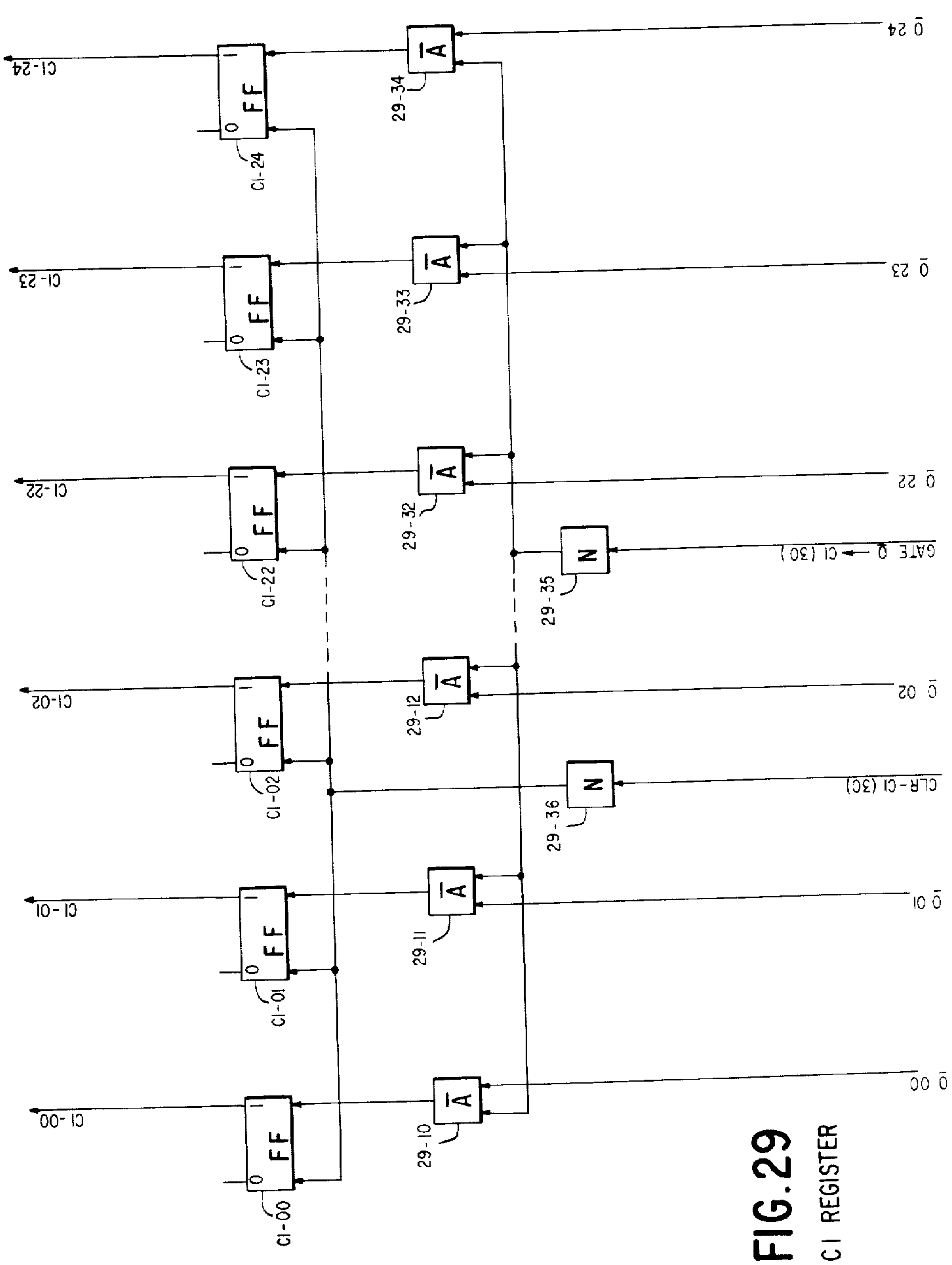


FIG.29

CI REGISTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 53

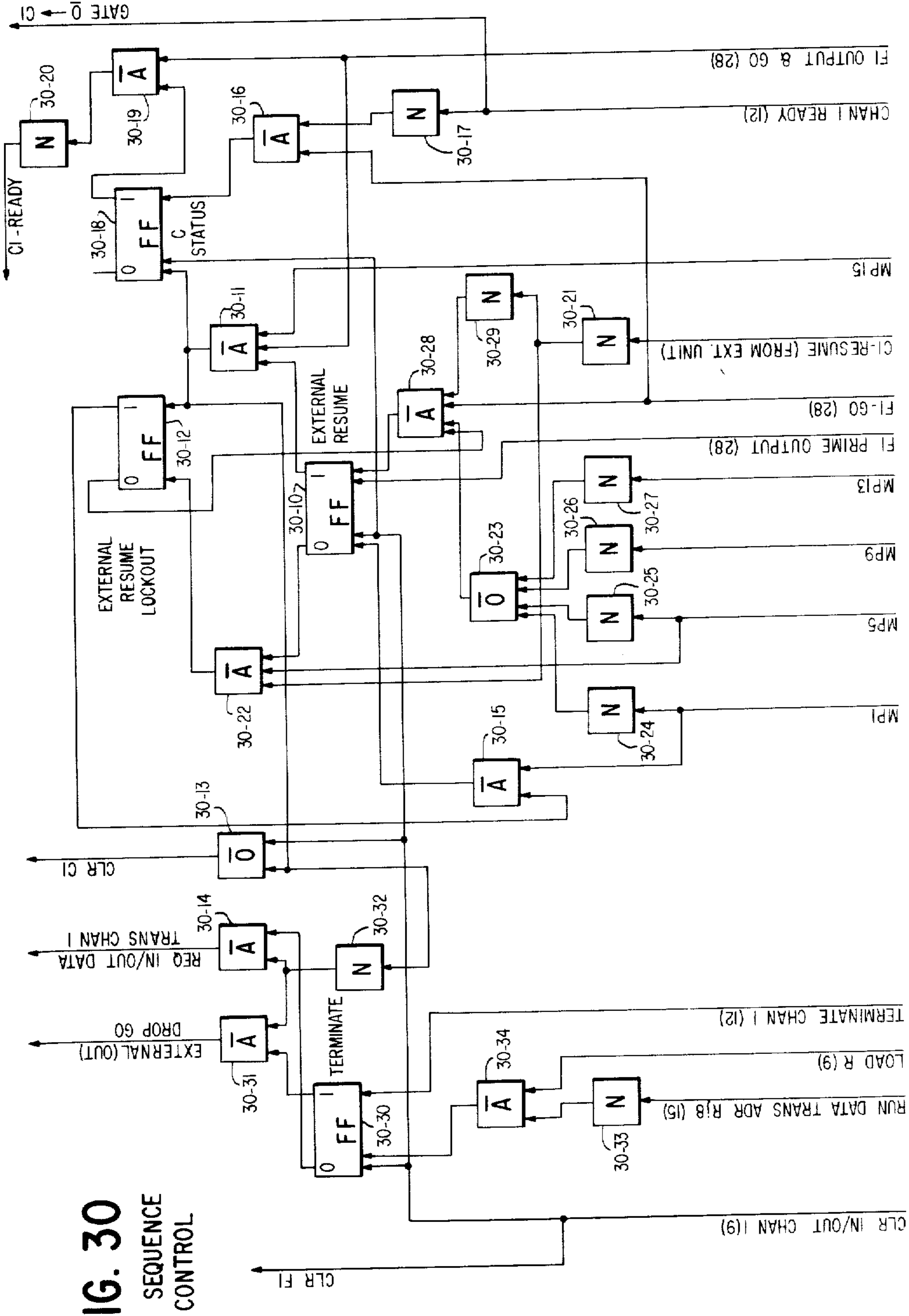


FIG. 30

CI SEQUENCE CONTROL

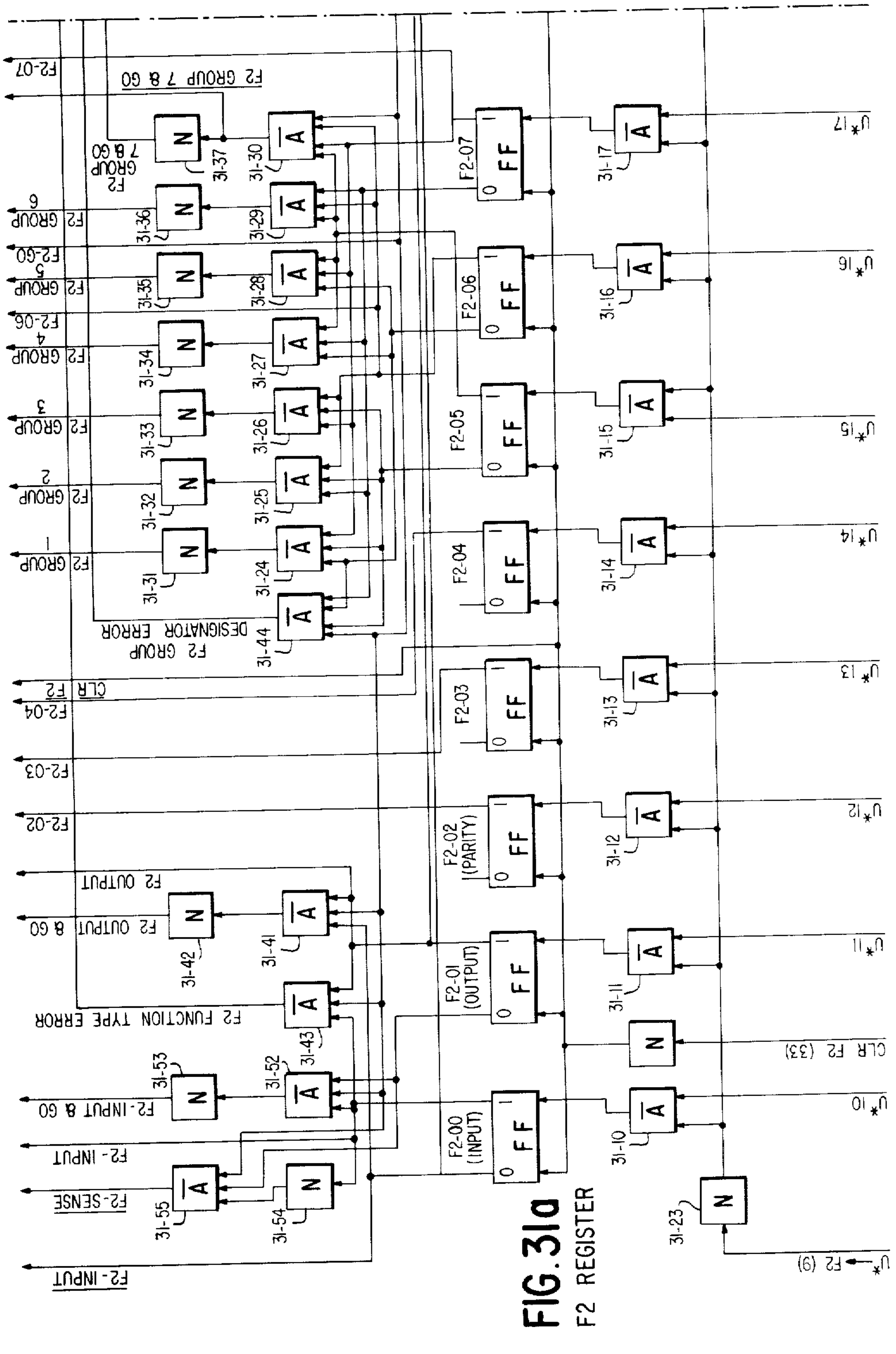


FIG. 31a
F2 REGISTER

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

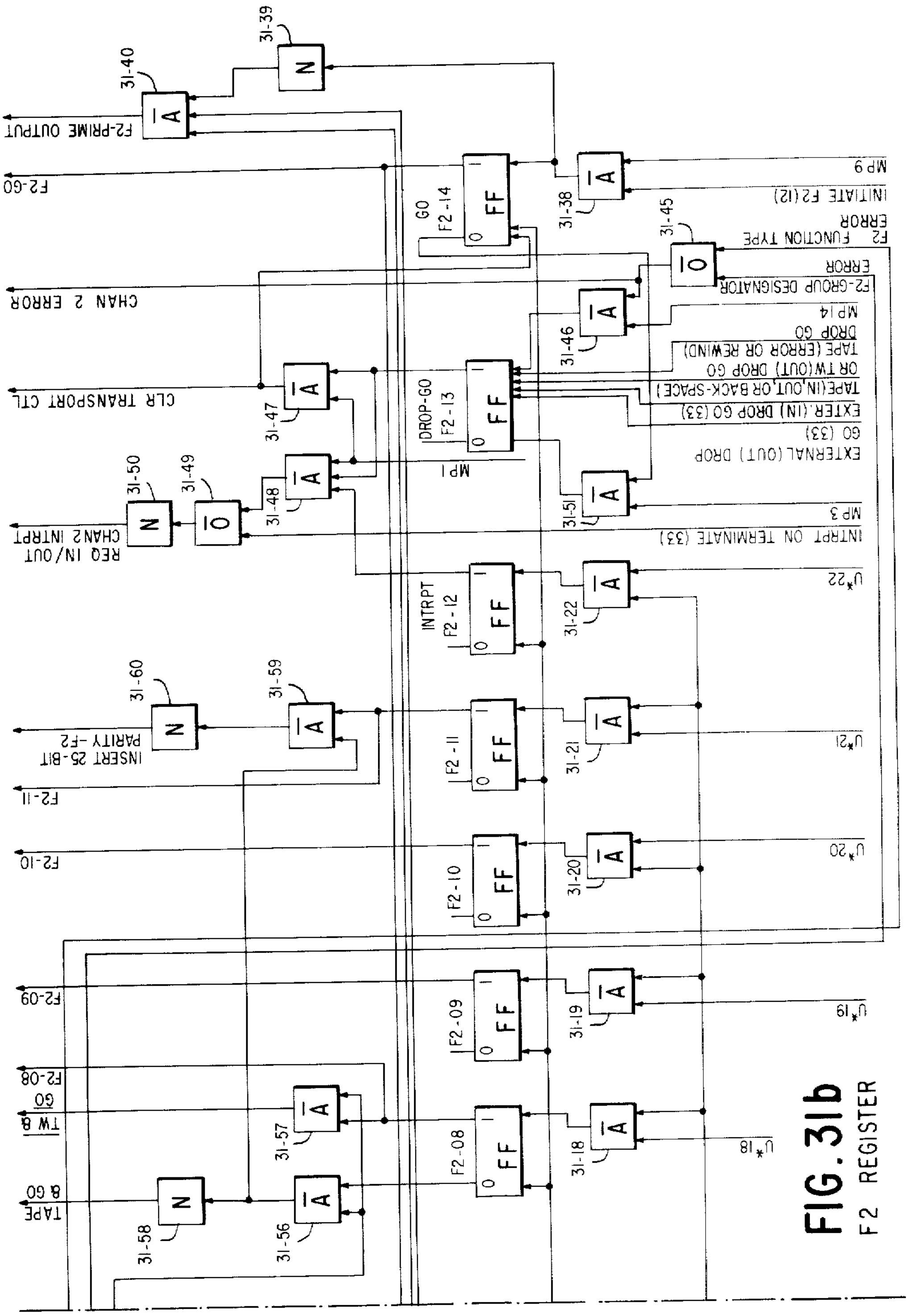


FIG. 31b
F2 REGISTER

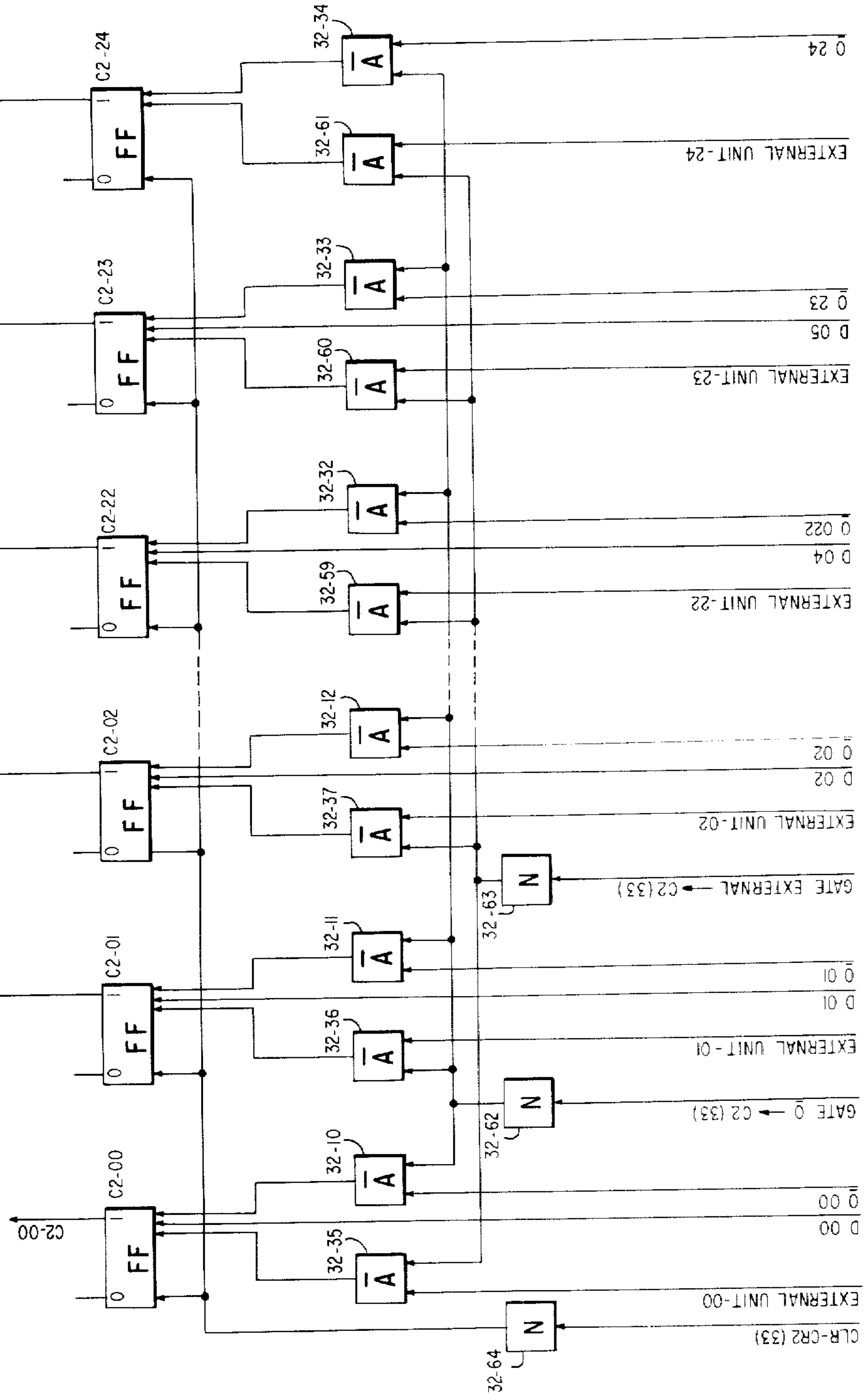


FIG.32
C2 REGISTER

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 57

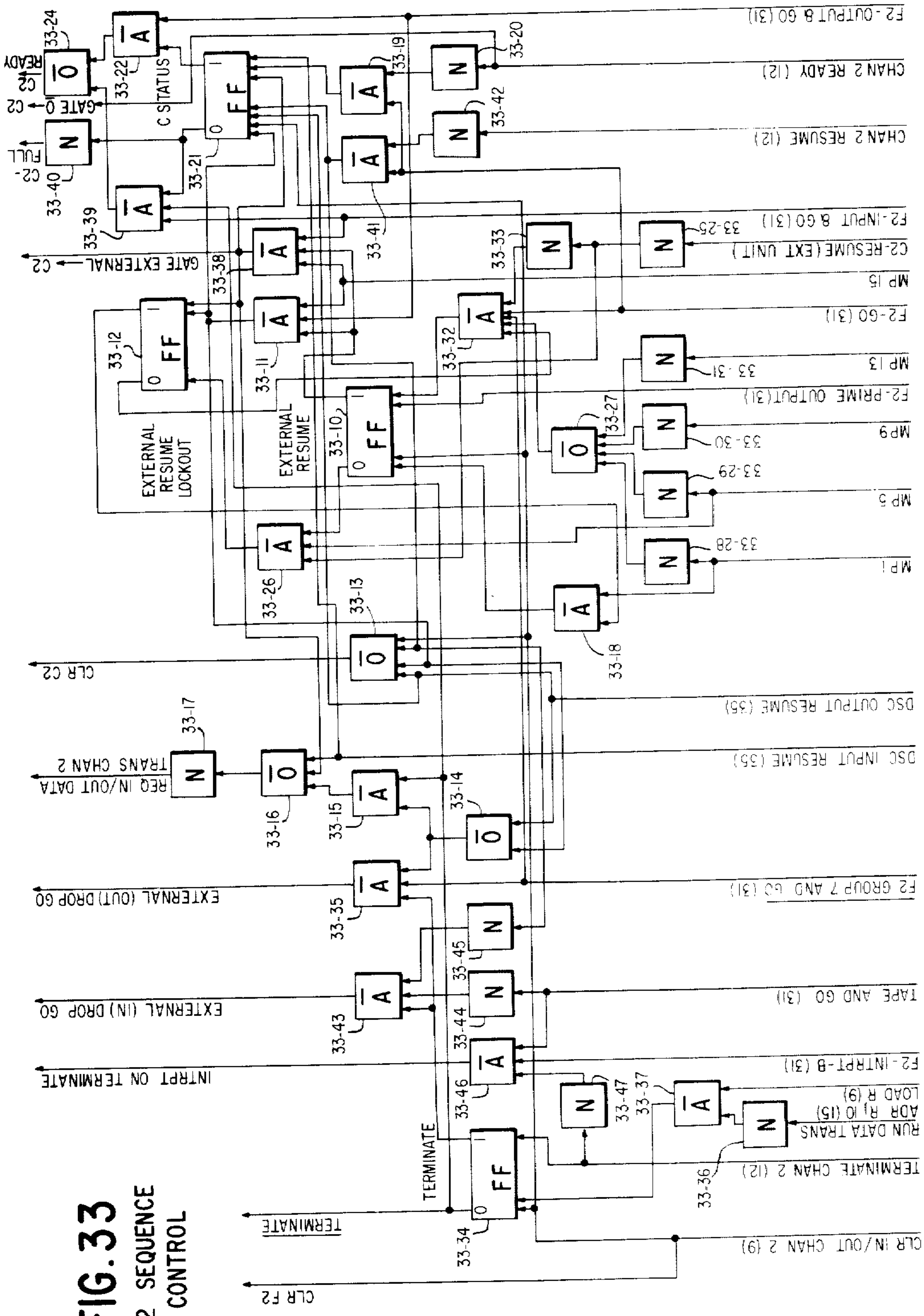


FIG. 33 C2 SEQUENCE CONTROL

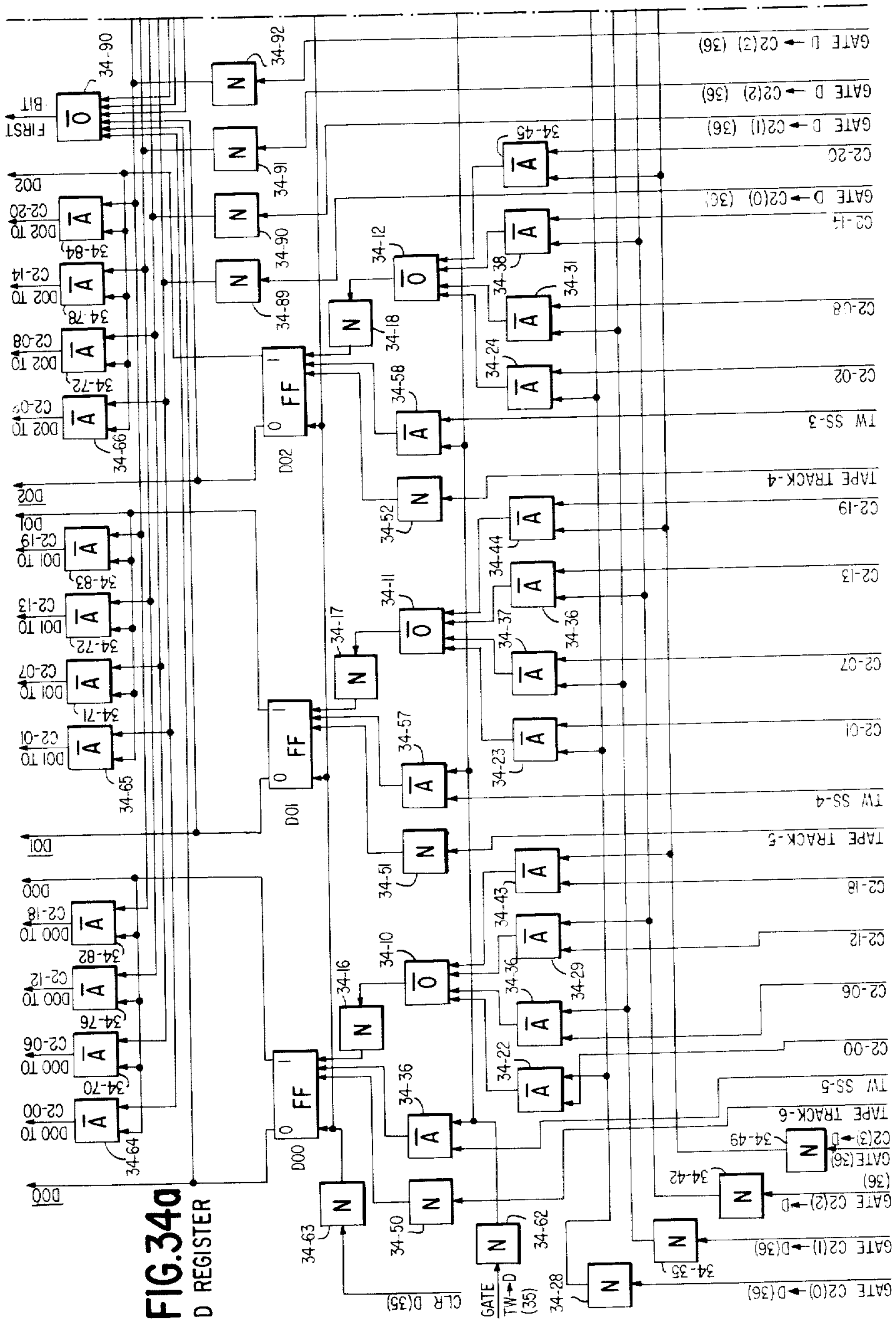


FIG. 34a
D REGISTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 59

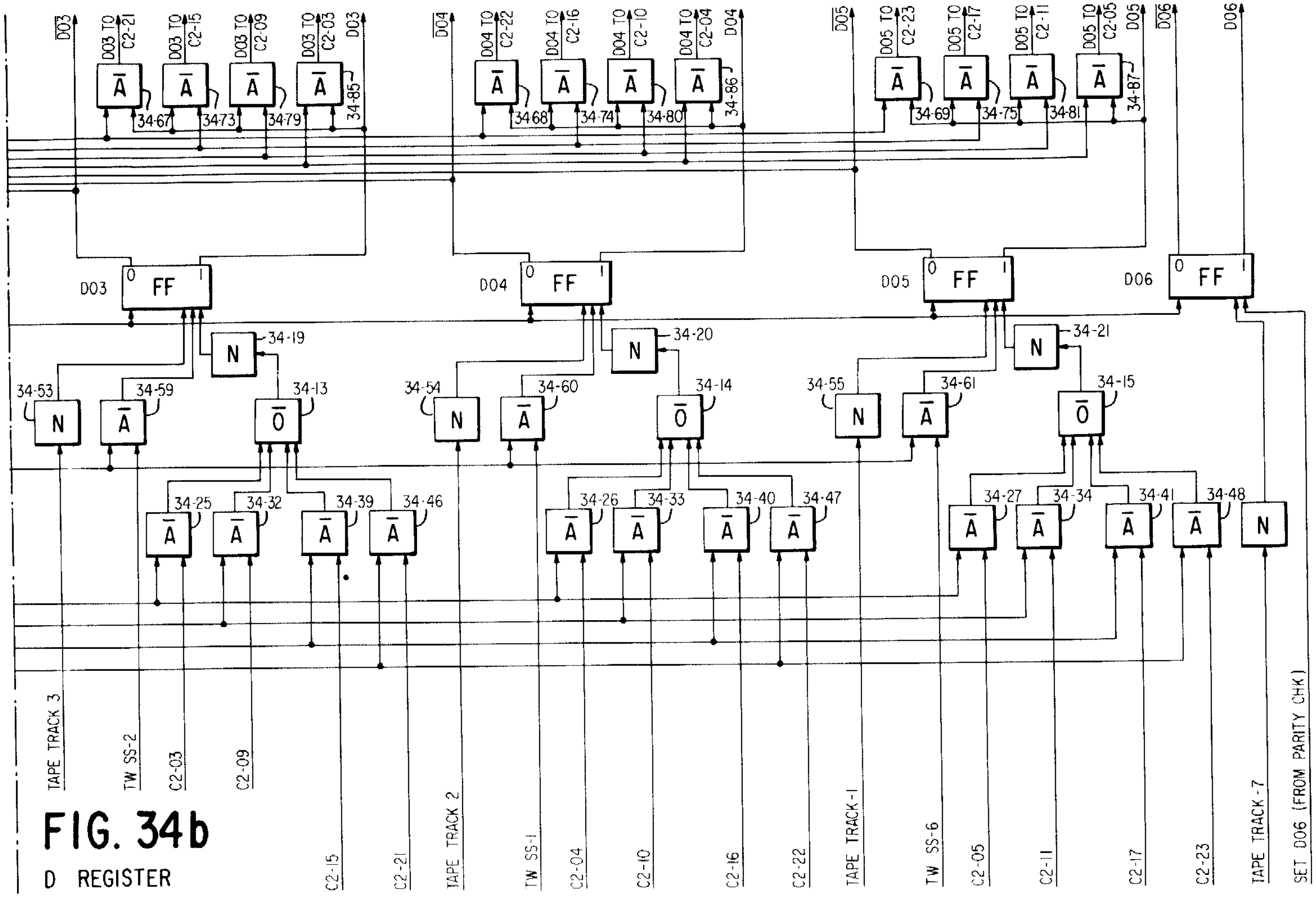


FIG. 34b

D REGISTER

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 60

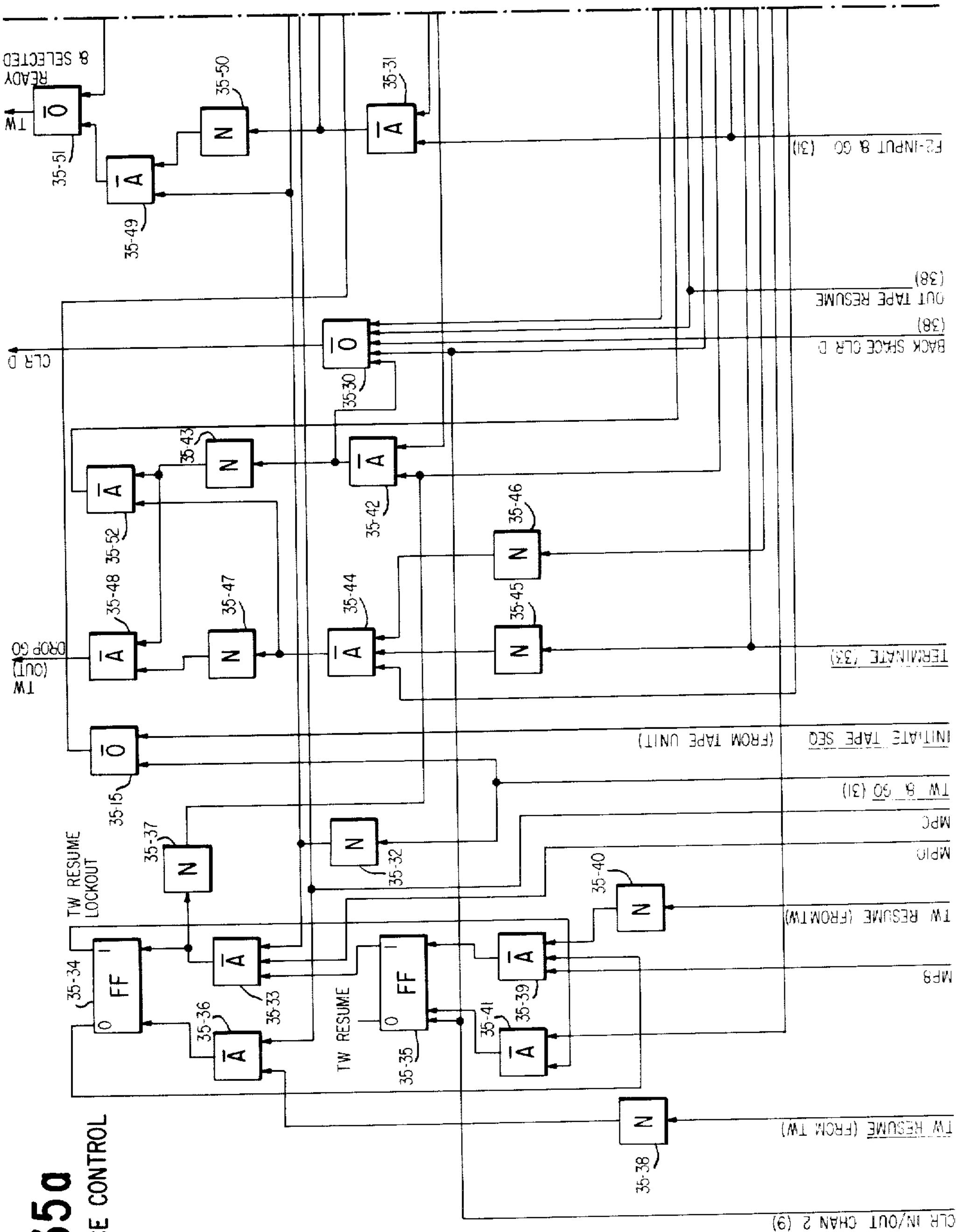


FIG. 35a

D SEQUENCE CONTROL

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

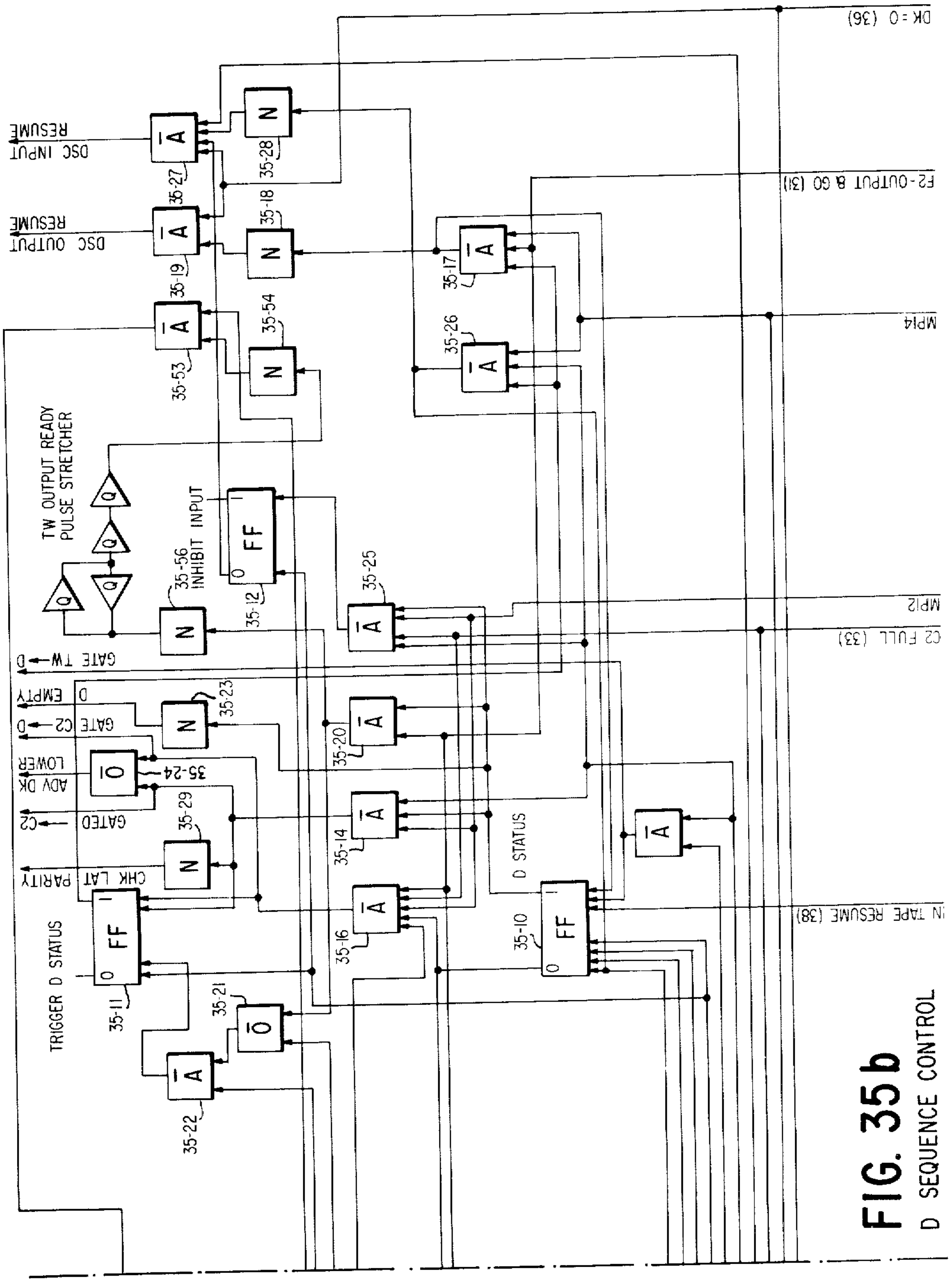


FIG. 35b

D SEQUENCE CONTROL

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 62

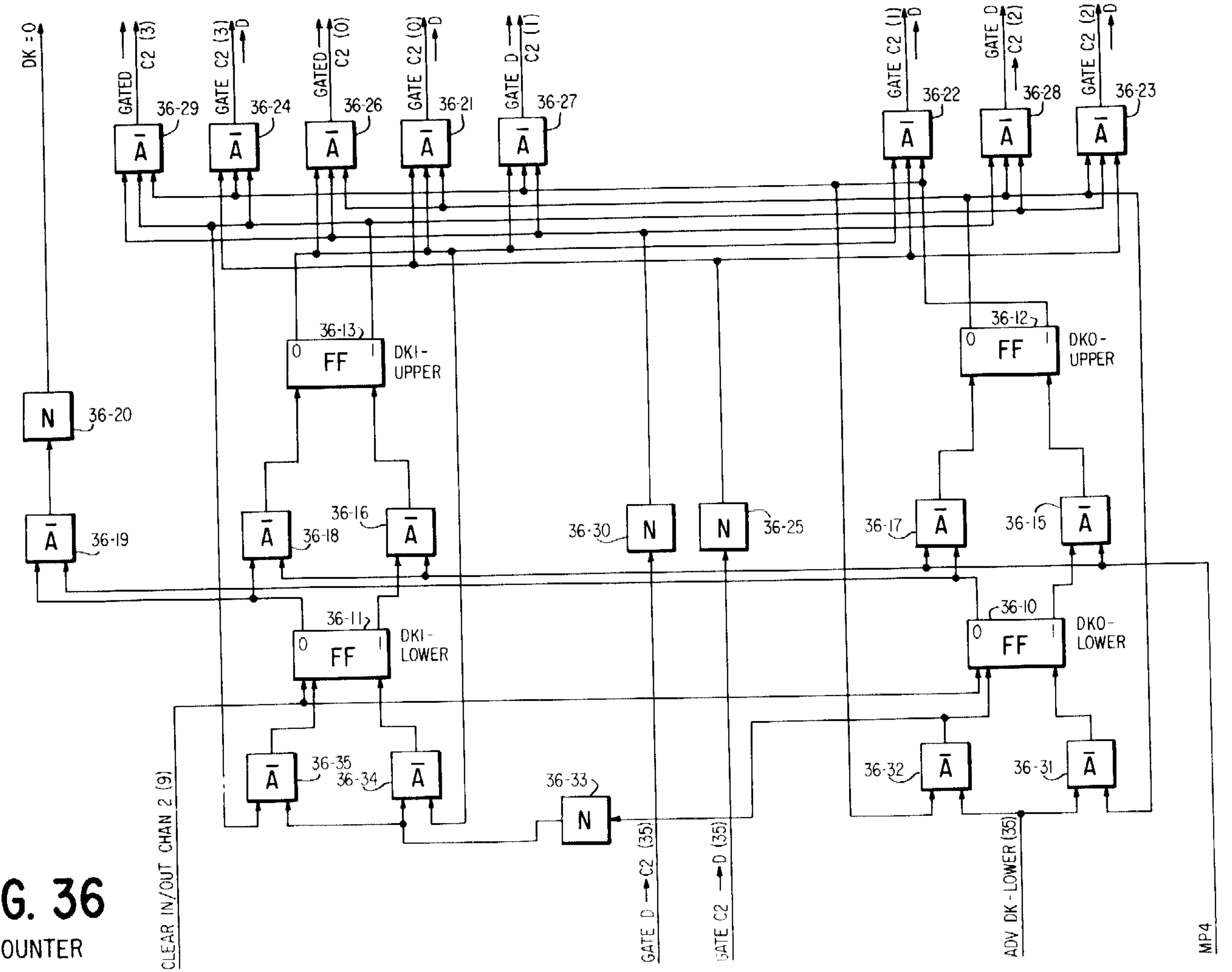


FIG. 36
D COUNTER

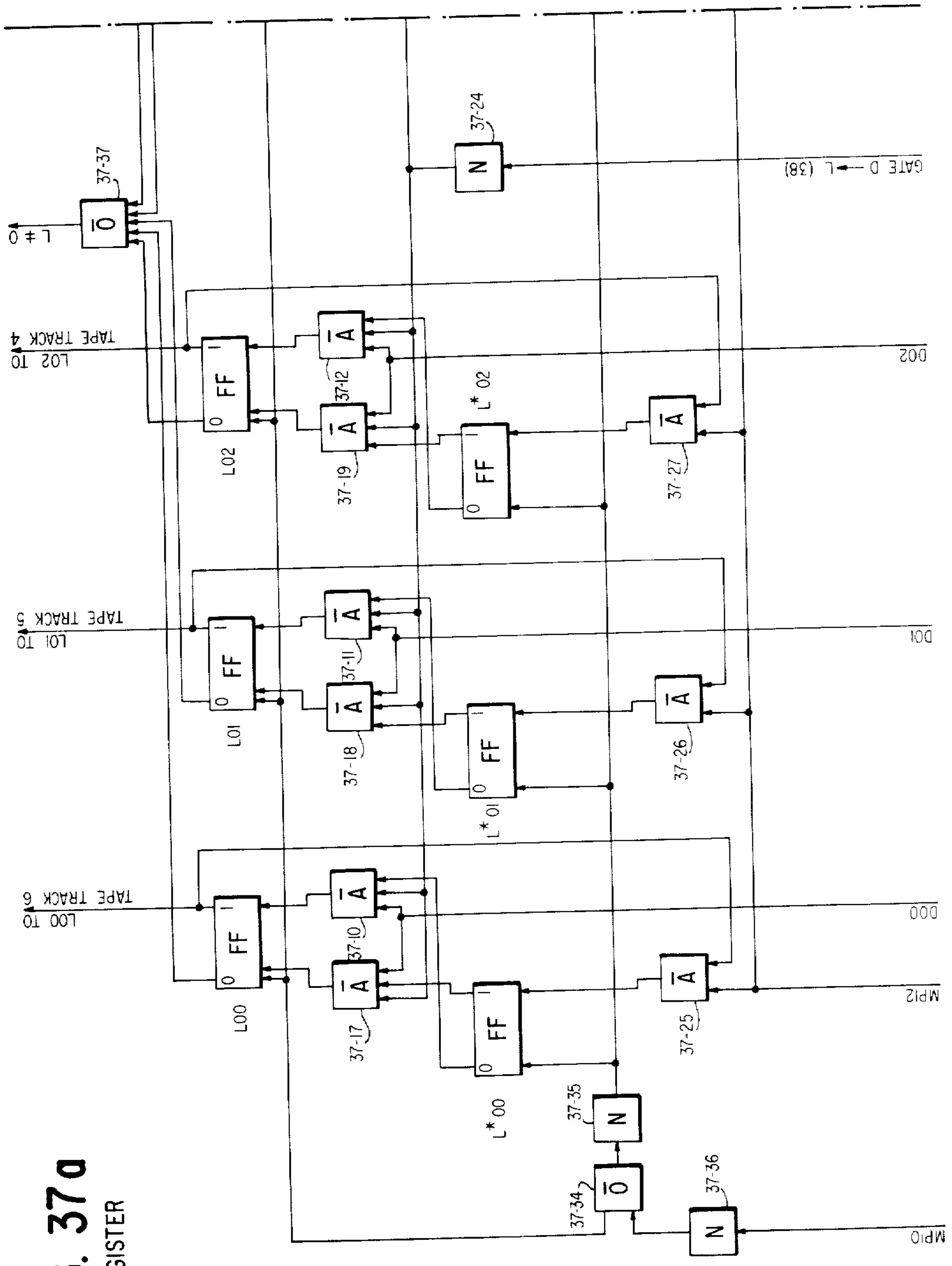


FIG. 37a
L REGISTER

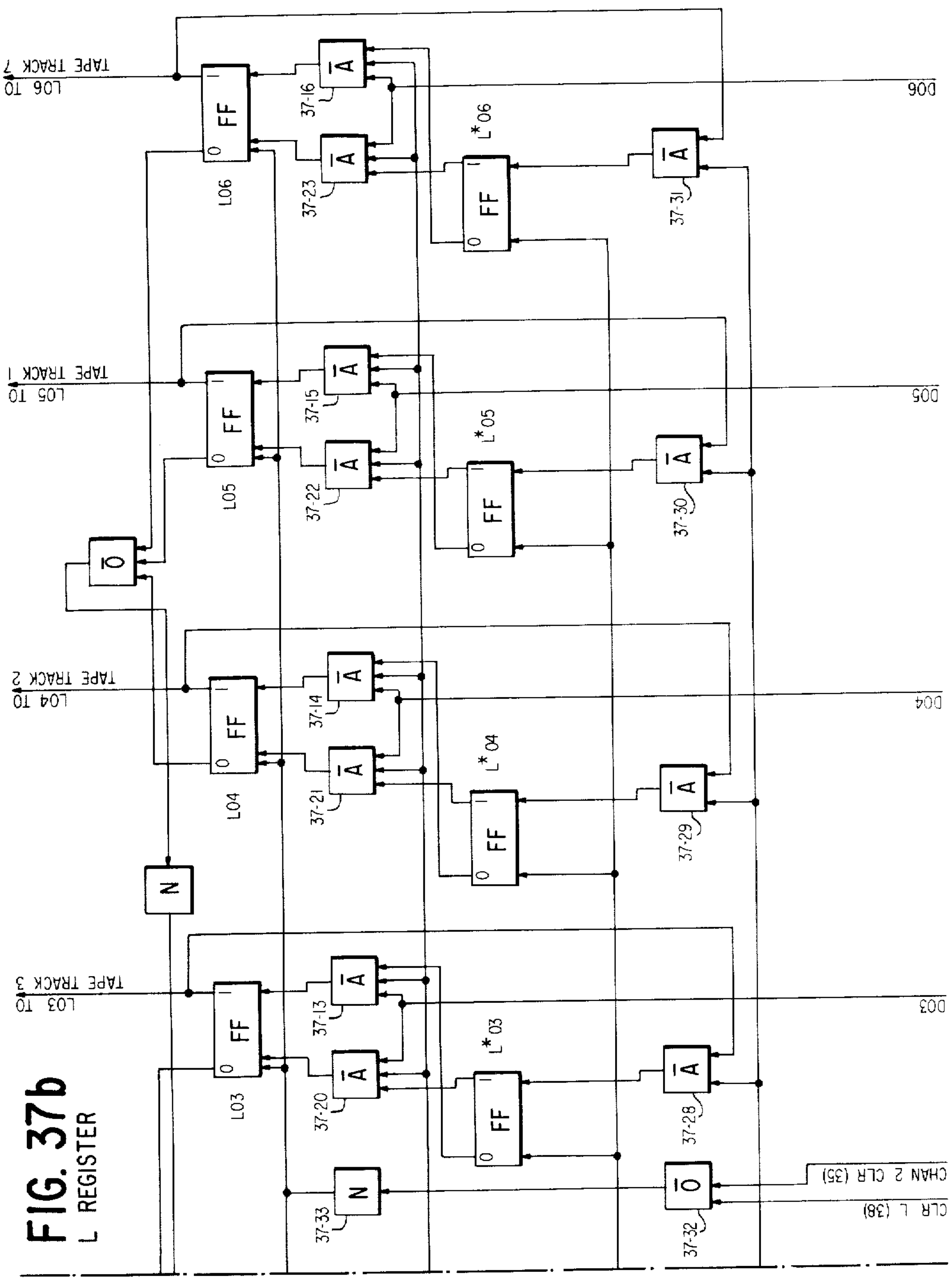


FIG. 37b
L REGISTER

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

93 Sheets-Sheet 65

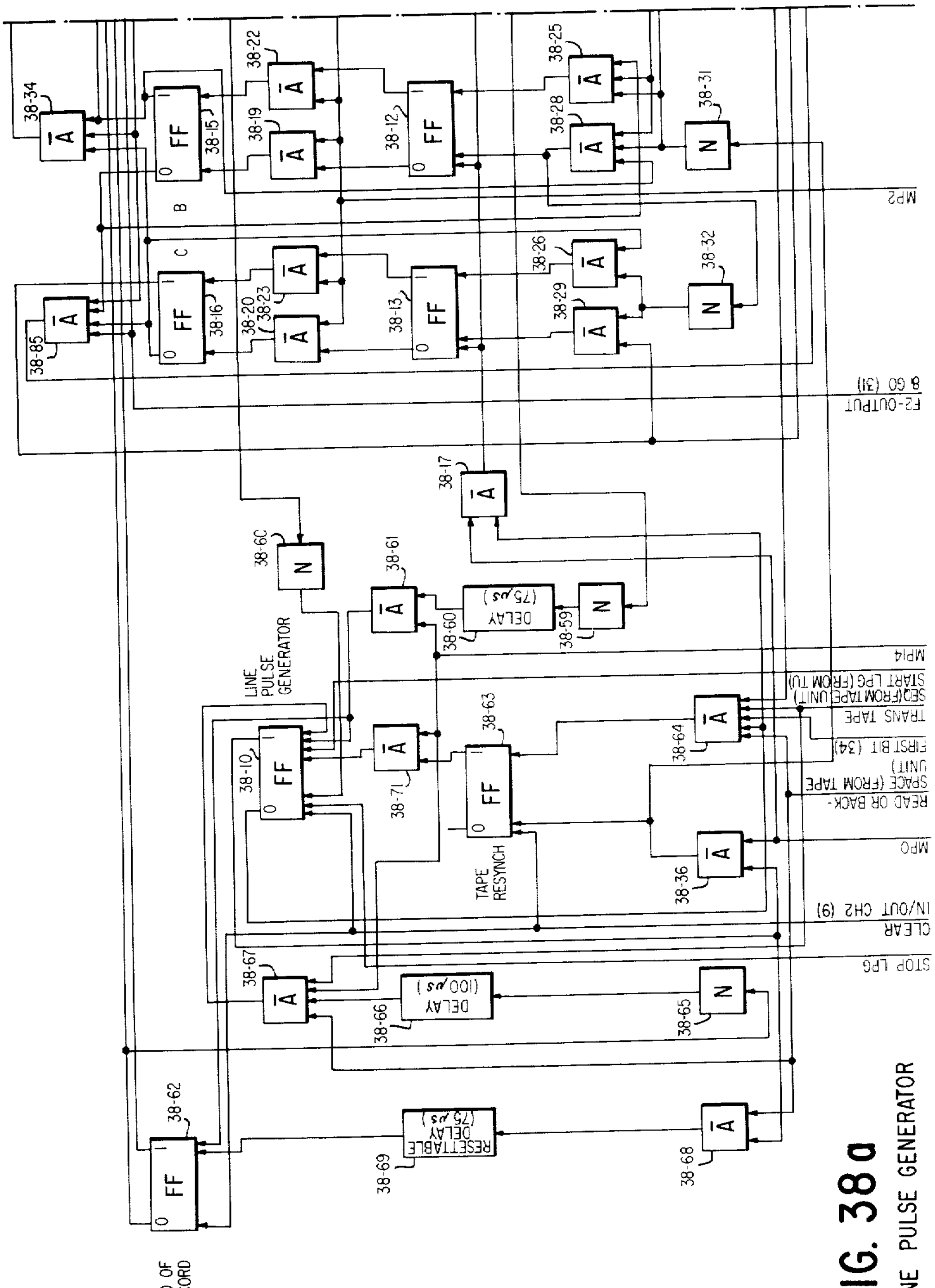


FIG. 38a
LINE PULSE GENERATOR

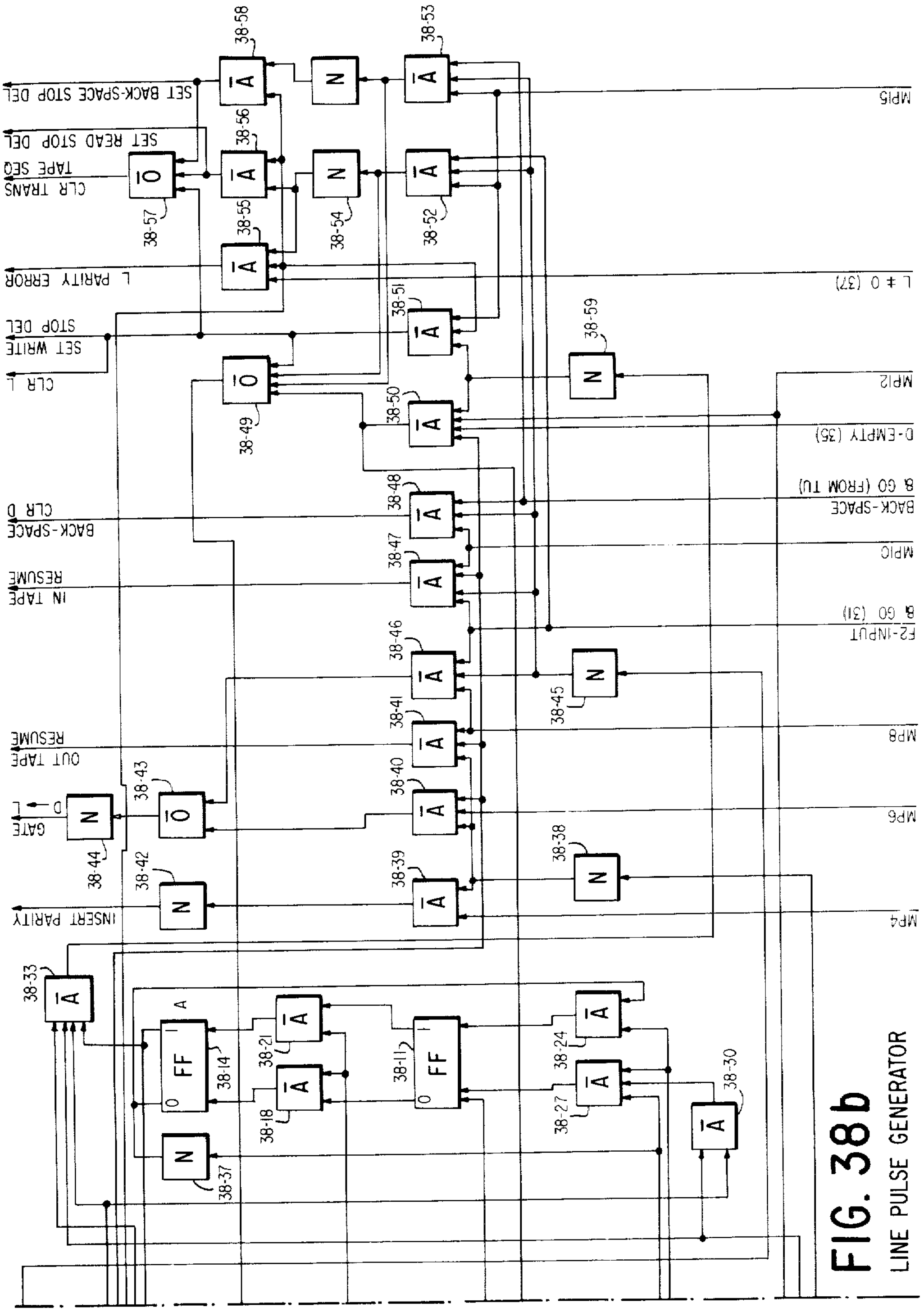


FIG. 38b
LINE PULSE GENERATOR

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 67

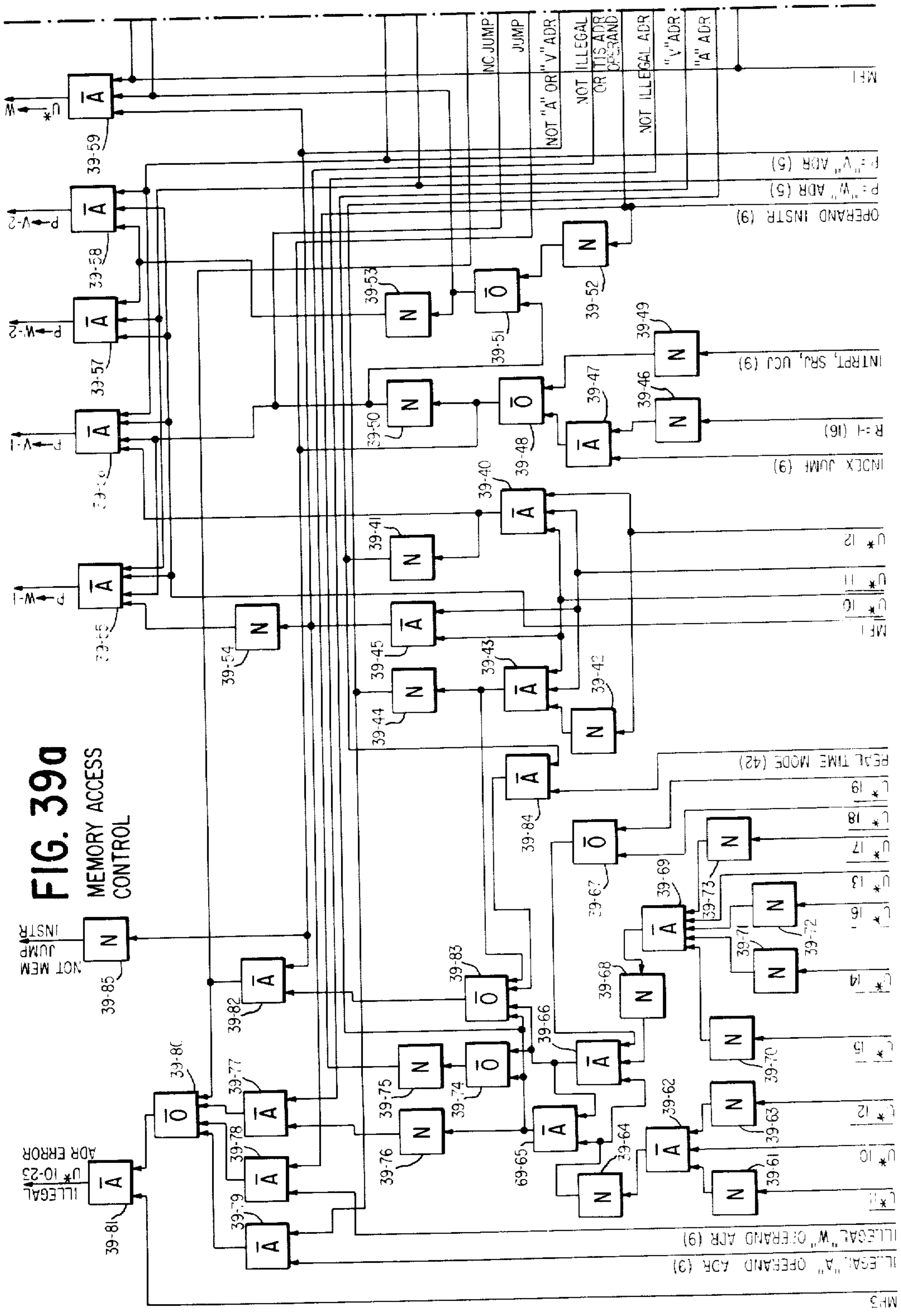


FIG. 39a
MEMORY ACCESS CONTROL

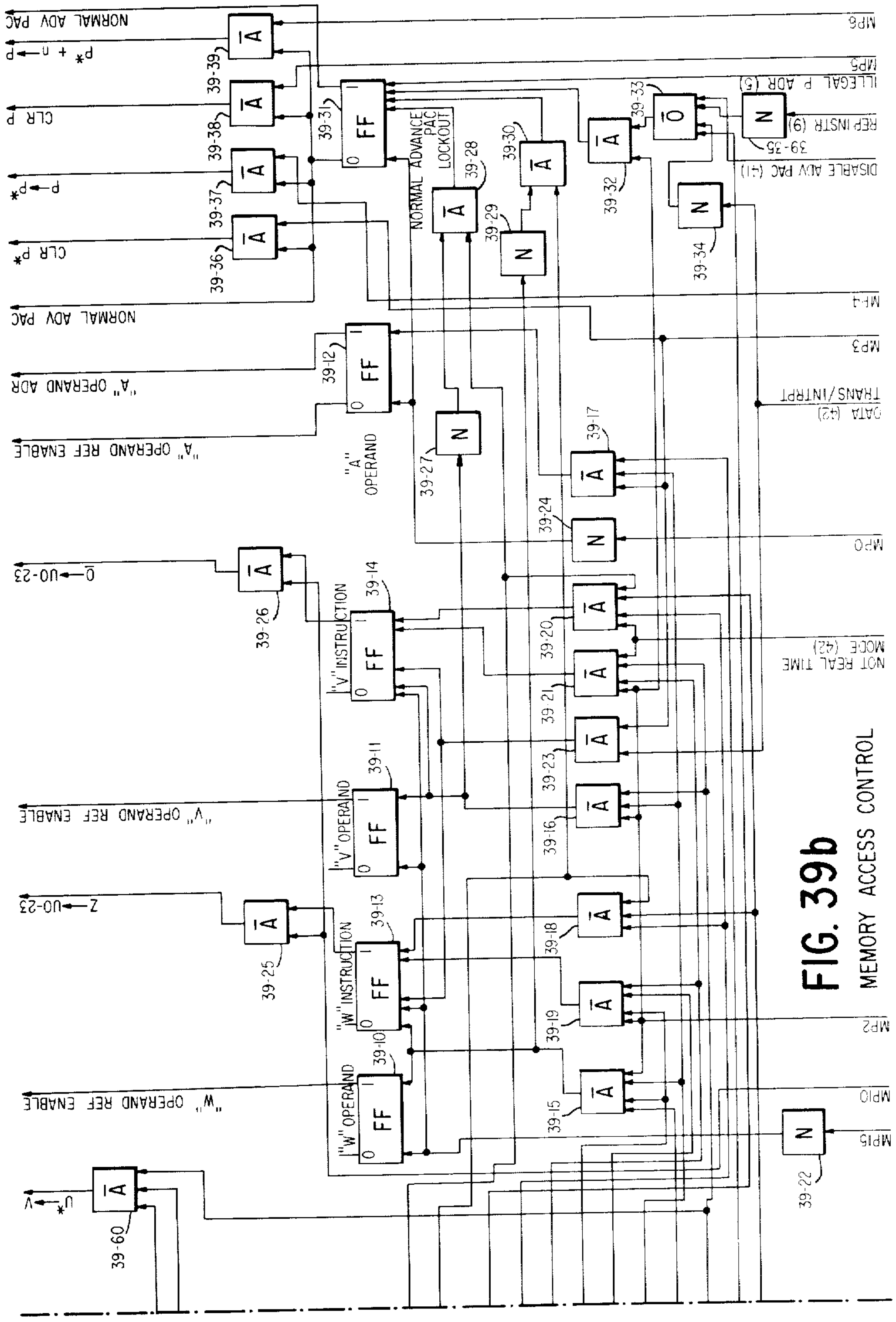


FIG. 39b MEMORY ACCESS CONTROL

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 69

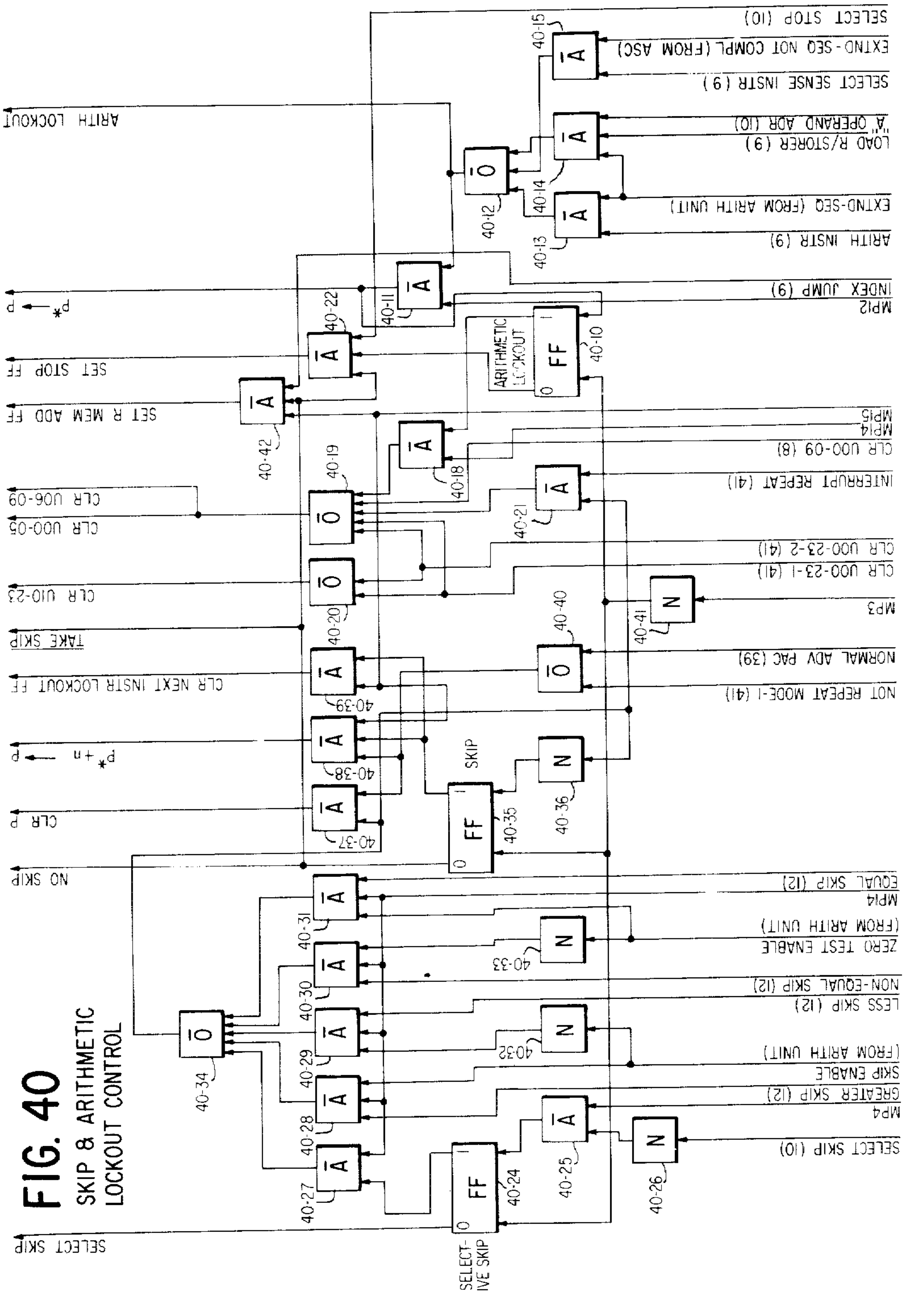


FIG. 40

SKIP & ARITHMETIC LOCKOUT CONTROL

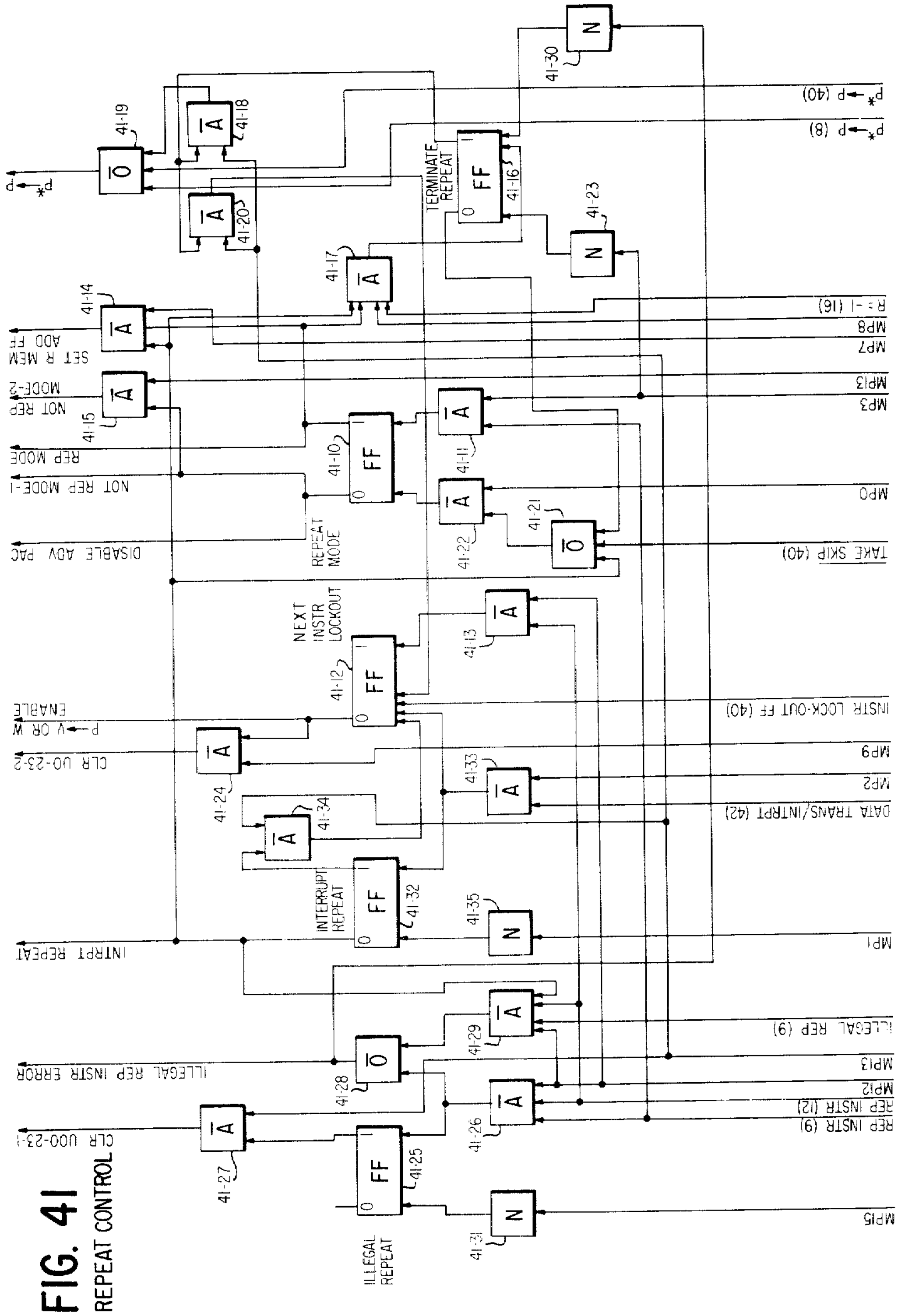


FIG. 41

REPEAT CONTROL

July 12, 1966 R. B. ARNDT ET AL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 71

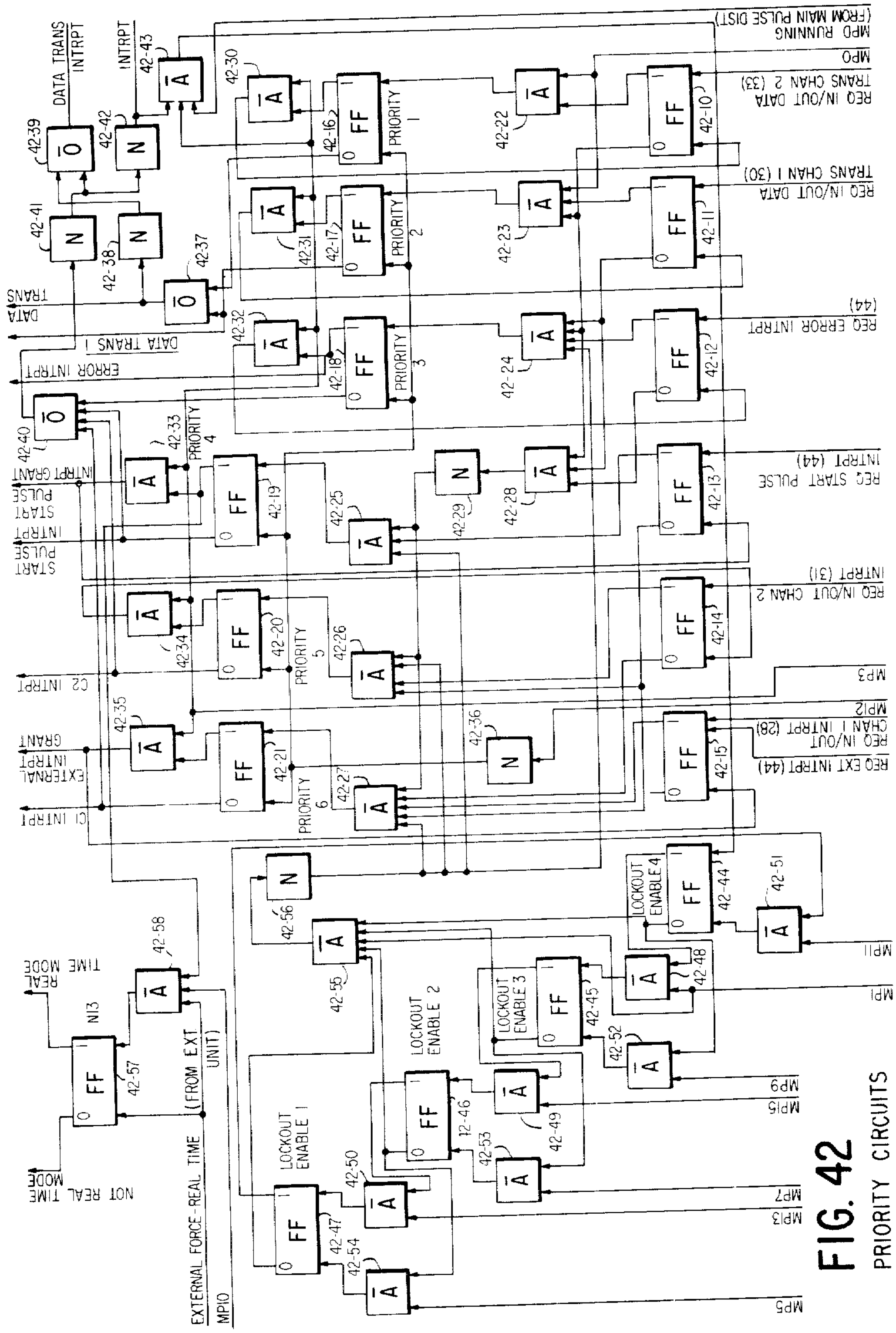


FIG. 42
PRIORITY CIRCUITS

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 72

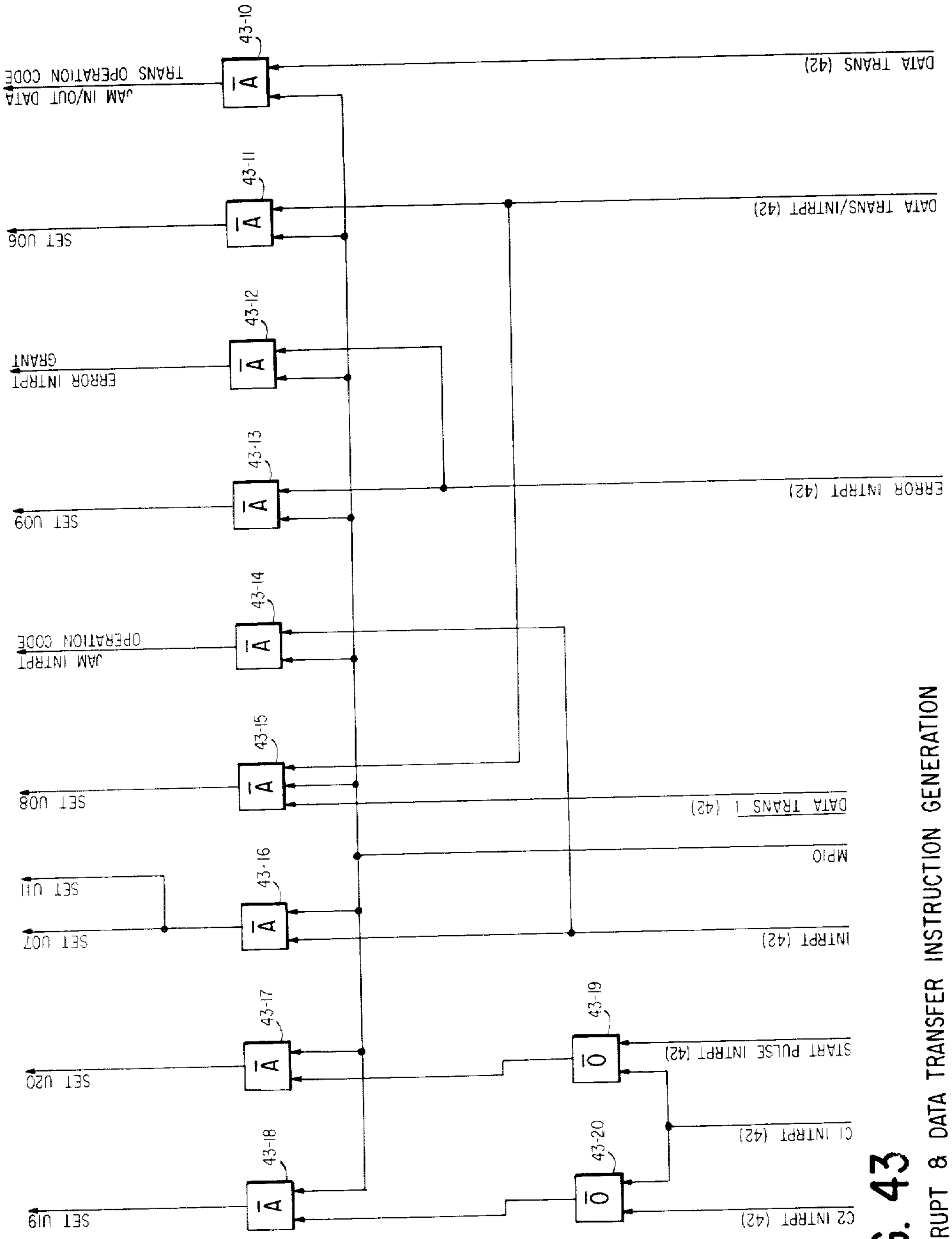


FIG. 43

INTERRUPT & DATA TRANSFER INSTRUCTION GENERATION

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 73

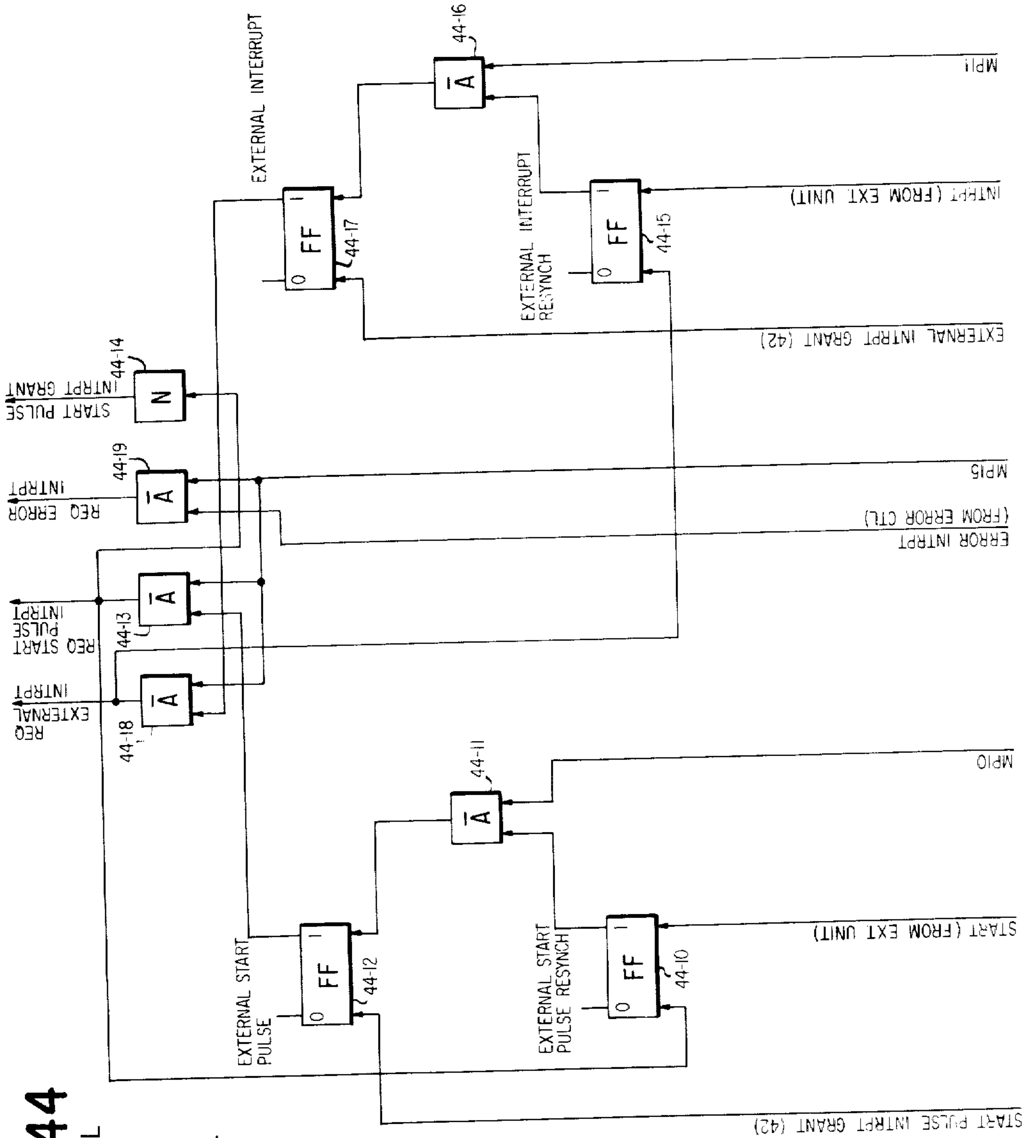


FIG. 44
EXTERNAL REAL TIME SYNCH. CONTROL

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 74

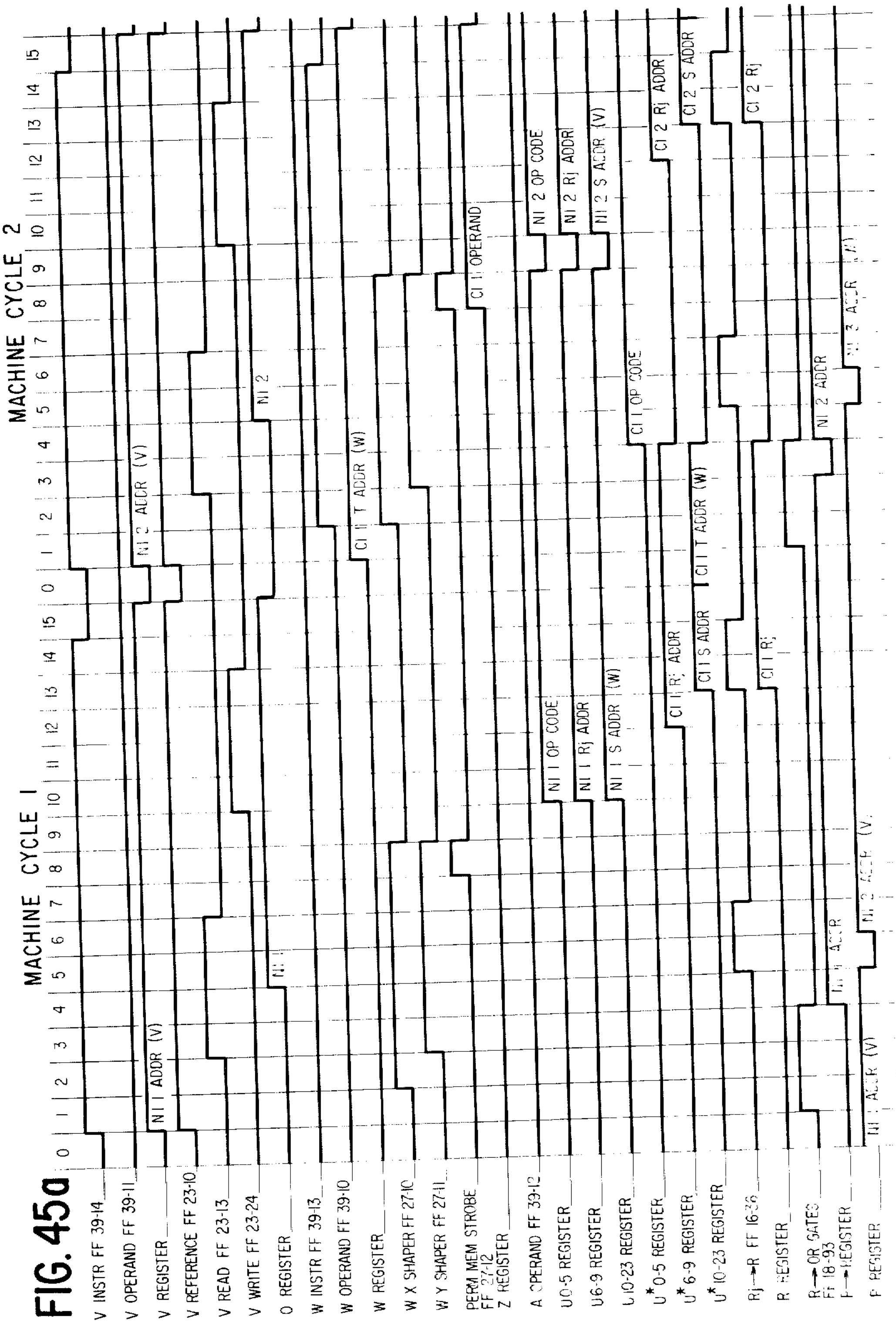


FIG. 45a

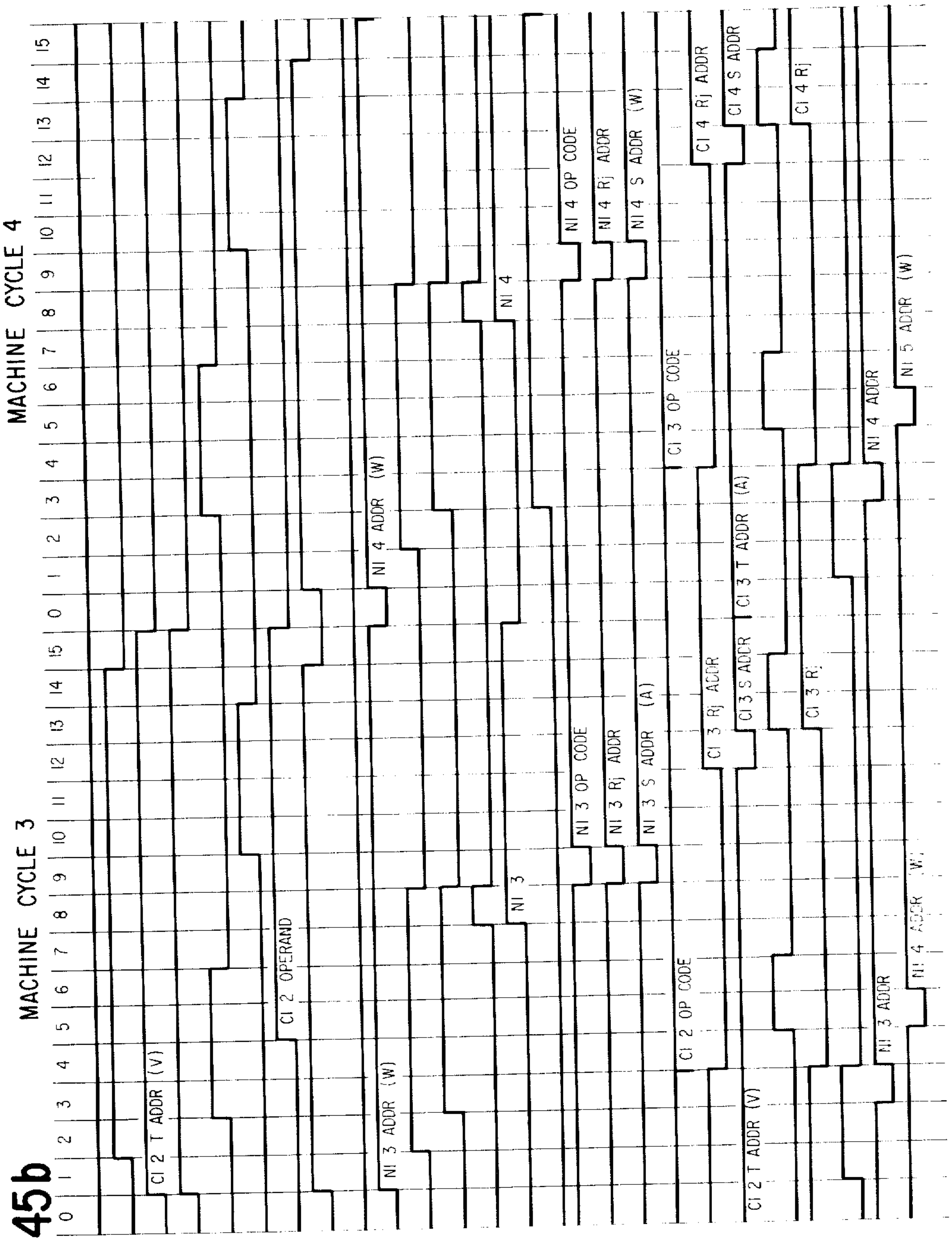

MACHINE CYCLE 3
MACHINE CYCLE 4
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
CI 2 T ADDR (V)
CI 2 OPERAND
NI 3 ADDR (W)
NI 4 ADDR (W)
NI 3
NI 4
NI 3 OP CODE
NI 3 Rj ADDR
NI 3 S ADDR (A)
NI 4 OP CODE
NI 4 Rj ADDR
NI 4 S ADDR (W)
CI 2 OP CODE
CI 3 OP CODE
CI 3 Rj ADDR
CI 4 Rj ADDR
CI 2 T ADDR (V)
CI 3 S ADDR
CI 3 T ADDR (A)
CI 4 S ADDR
CI 3 Rj
CI 4 Rj
NI 3 ADDR
NI 4 ADDR
NI 4 ADDR (W)
NI 5 ADDR (W)

FIG. 45b

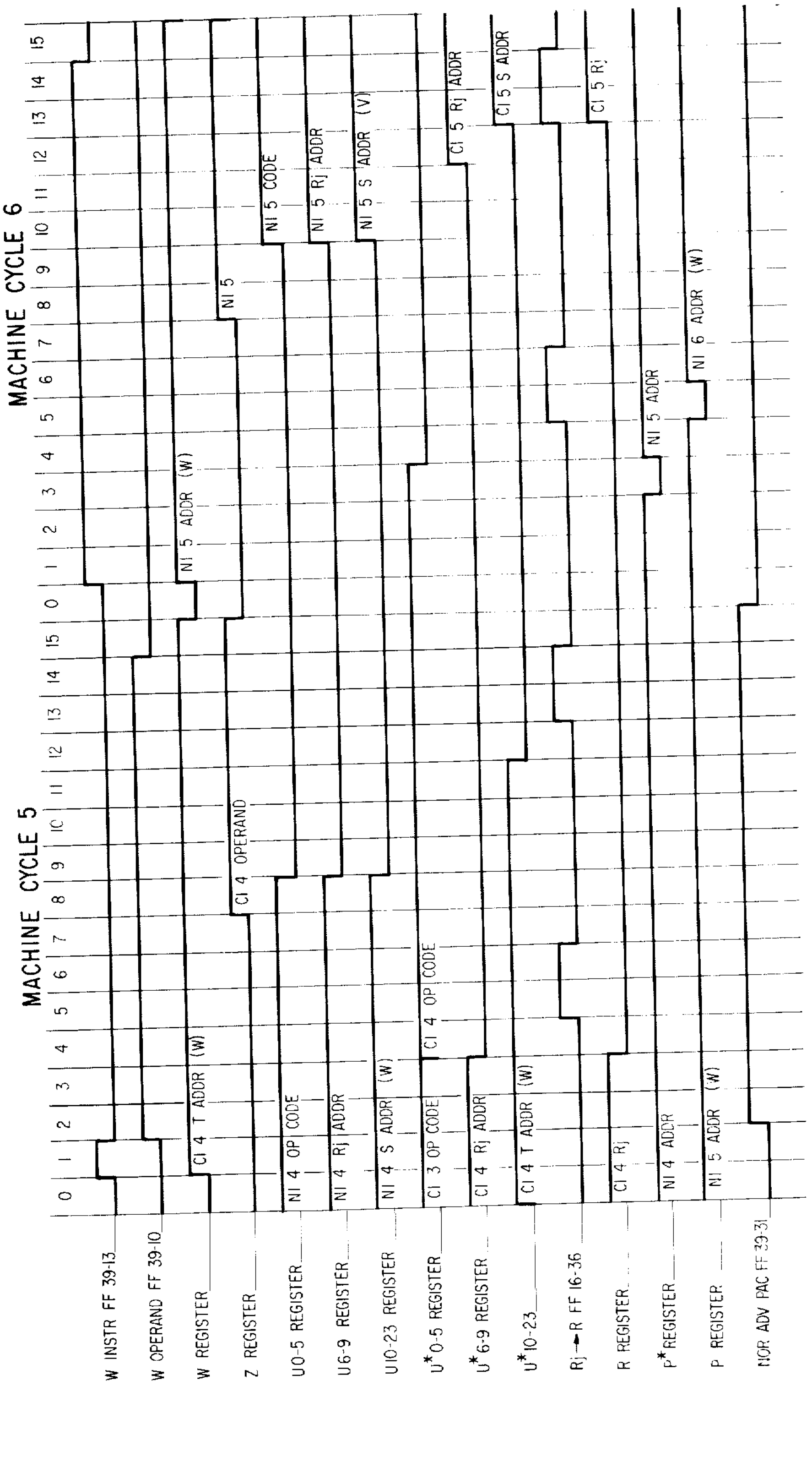

MACHINE CYCLE 5
MACHINE CYCLE 6
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
W INSTR FF 39-13
W OPERAND FF 39-10
W REGISTER
Z REGISTER
U0-5 REGISTER
U6-9 REGISTER
U10-23 REGISTER
U*0-5 REGISTER
U*6-9 REGISTER
U*10-23
Rj→R FF 16-36
R REGISTER
P*REGISTER
P REGISTER
NOR ADV PAC FF 39-31
CI 4 T ADDR (W)
CI 4 OPERAND
NI 4 OP CODE
NI 4 Rj ADDR
NI 4 S ADDR (W)
CI 3 OP CODE
CI 4 OP CODE
CI 4 Rj ADDR
CI 4 T ADDR (W)
CI 4 Rj
NI 4 ADDR
NI 5 ADDR (W)
NI 5 ADDR (W)
NI 5
NI 5 CODE
NI 5 Rj ADDR
NI 5 S ADDR (V)
CI 5 Rj ADDR
CI 5 S ADDR
CI 5 Rj
NI 5 ADDR
NI 6 ADDR (W)

FIG. 46

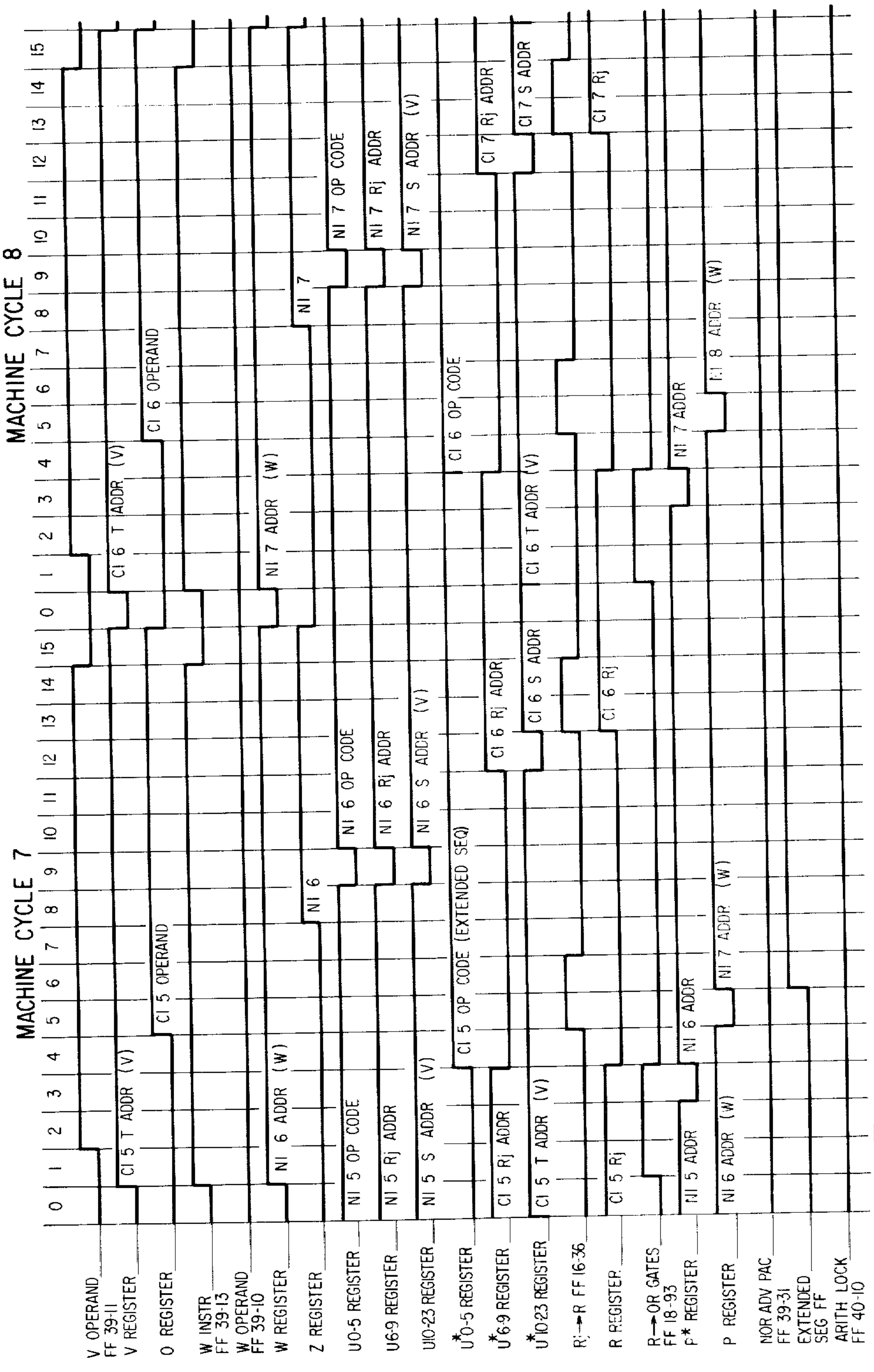

MACHINE CYCLE 7
MACHINE CYCLE 8
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
V OPERAND FF 39-11
V REGISTER
O REGISTER
W INSTR FF 39-13
W OPERAND FF 39-10
W REGISTER
Z REGISTER
U0-5 REGISTER
U6-9 REGISTER
U10-23 REGISTER
*U0-5 REGISTER
*U6-9 REGISTER
*U10-23 REGISTER
Rj→R FF 16-36
R REGISTER
R→OR GATES FF 18-93
P* REGISTER
P REGISTER
NOR ADV PAC FF 39-31
EXTENDED SEG FF
ARITH LOCK FF 40-10
CI 5 T ADDR (V)
CI 5 OPERAND
NI 6 ADDR (W)
NI 6
NI 5 OP CODE
NI 5 Rj ADDR
NI 5 S ADDR (V)
NI 6 OP CODE
NI 6 Rj ADDR
NI 6 S ADDR (V)
CI 5 OP CODE (EXTENDED SEQ)
CI 5 Rj ADDR
CI 5 T ADDR (V)
CI 6 Rj ADDR
CI 6 S ADDR
CI 5 Rj
CI 6 Rj
NI 5 ADDR
NI 6 ADDR
NI 6 ADDR (W)
NI 7 ADDR (W)
CI 6 T ADDR (V)
CI 6 OPERAND
NI 7 ADDR (W)
NI 7
NI 7 OP CODE
NI 7 Rj ADDR
NI 7 S ADDR (V)
CI 6 OP CODE
CI 7 Rj ADDR
CI 7 S ADDR
CI 6 T ADDR (V)
CI 7 Rj
NI 7 ADDR
NI 8 ADDR (W)

FIG. 47a

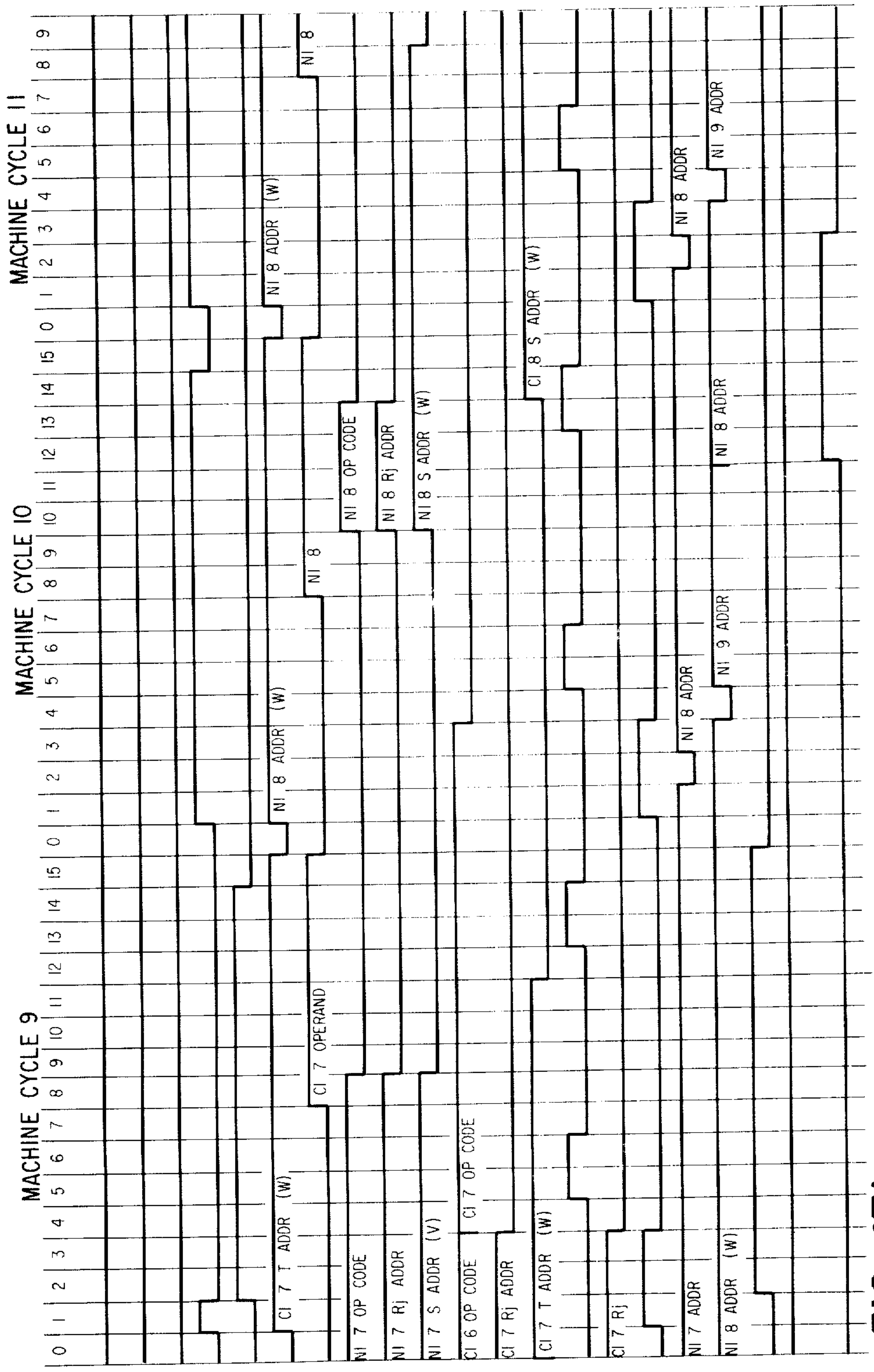

MACHINE CYCLE 9
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
MACHINE CYCLE 10
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
MACHINE CYCLE 11
0
1
2
3
4
5
6
7
8
9
CI 7 T ADDR (W)
NI 8 ADDR (W)
NI 8 ADDR (W)
CI 7 OPERAND
NI 8
NI 8
NI 7 OP CODE
NI 8 OP CODE
NI 7 Rj ADDR
NI 8 Rj ADDR
NI 7 S ADDR (V)
NI 8 S ADDR (W)
CI 6 OP CODE
CI 7 OP CODE
CI 7 Rj ADDR
CI 7 T ADDR (W)
CI 8 S ADDR (W)
CI 7 Rj
NI 7 ADDR
NI 8 ADDR
NI 8 ADDR
NI 8 ADDR (W)
NI 9 ADDR
NI 8 ADDR
NI 9 ADDR

FIG. 47b

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

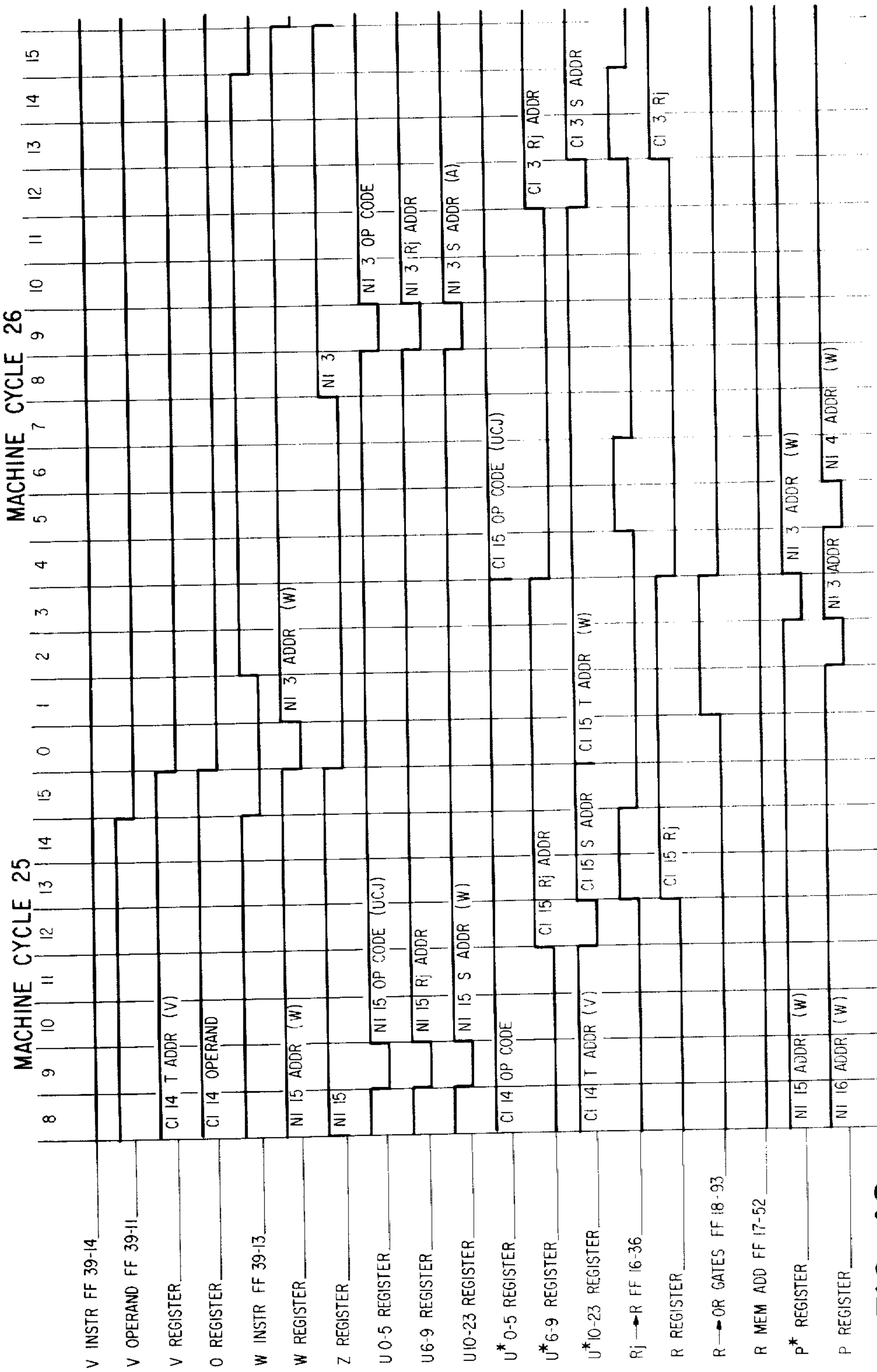


FIG. 48a

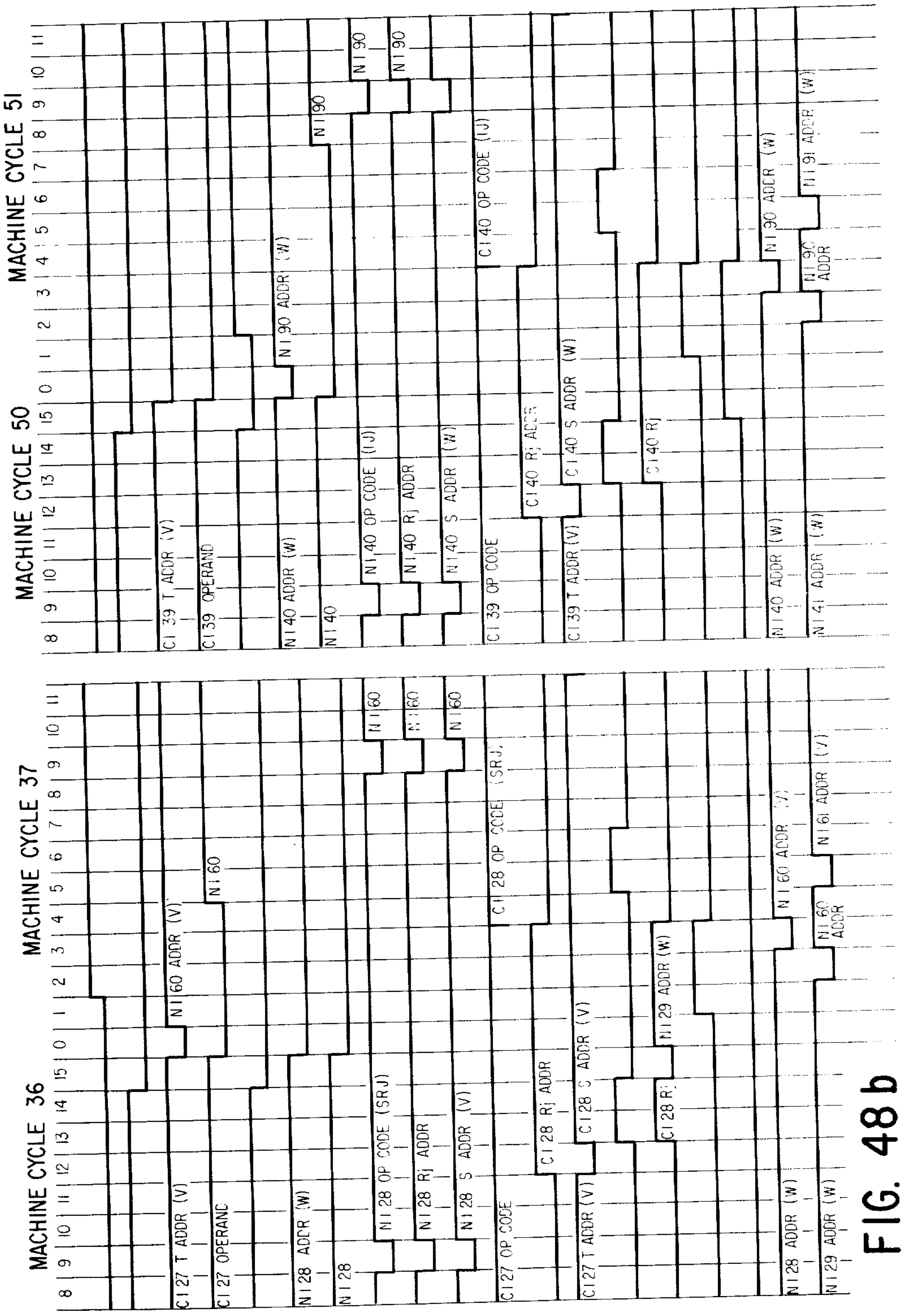

MACHINE CYCLE 50
MACHINE CYCLE 51
8 9 10 11 12 13 14 15 0 1 2 3 4 5 6 7 8 9 10 11
C I 39 T ADDR (V)
C I 39 OPERAND
N I 40 ADDR (W)
N I 90 ADDR (W)
N I 40
N I 90
N I 90
N I 90
N I 40 OP CODE (IJ)
N I 40 Rj ADDR
N I 40 S ADDR (W)
C I 39 OP CODE
C I 40 OP CODE (IJ)
C I 40 Rj ADDR
C I 39 T ADDR (V)
C I 40 S ADDR (W)
C I 40 Rj
N I 40 ADDR (W)
N I 90 ADDR (W)
N I 41 ADDR (W)
N I 90 ADDR
N I 91 ADDR (W)
MACHINE CYCLE 36
MACHINE CYCLE 37
8 9 10 11 12 13 14 15 0 1 2 3 4 5 6 7 8 9 10 11
C I 27 T ADDR (V)
N I 60 ADDR (V)
C I 27 OPERAND
N I 60
N I 28 ADDR (W)
N I 28
N I 28 OP CODE (SRJ)
N I 60
N I 28 Rj ADDR
N I 60
N I 28 S ADDR (V)
N I 60
C I 27 OP CODE
C I 28 OP CODE (SRJ)
C I 28 Rj ADDR
C I 27 T ADDR (V)
C I 28 S ADDR (V)
C I 28 Rj
N I 29 ADDR (W)
N I 28 ADDR (W)
N I 60 ADDR (V)
N I 29 ADDR (W)
N I 60 ADDR
N I 61 ADDR (V)

FIG. 48b

July 12, 1966 R. B. ARNDT ET AL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 81

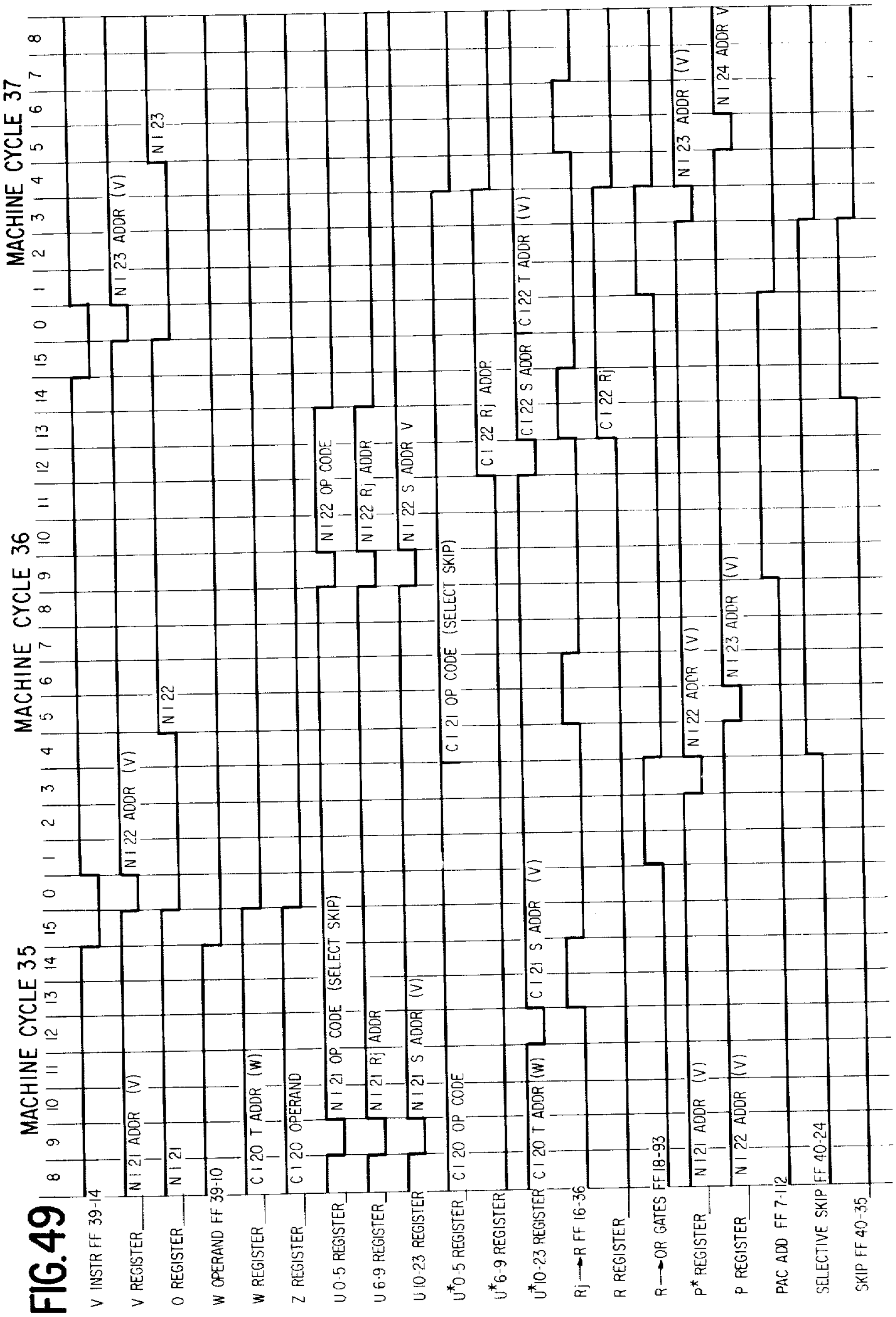


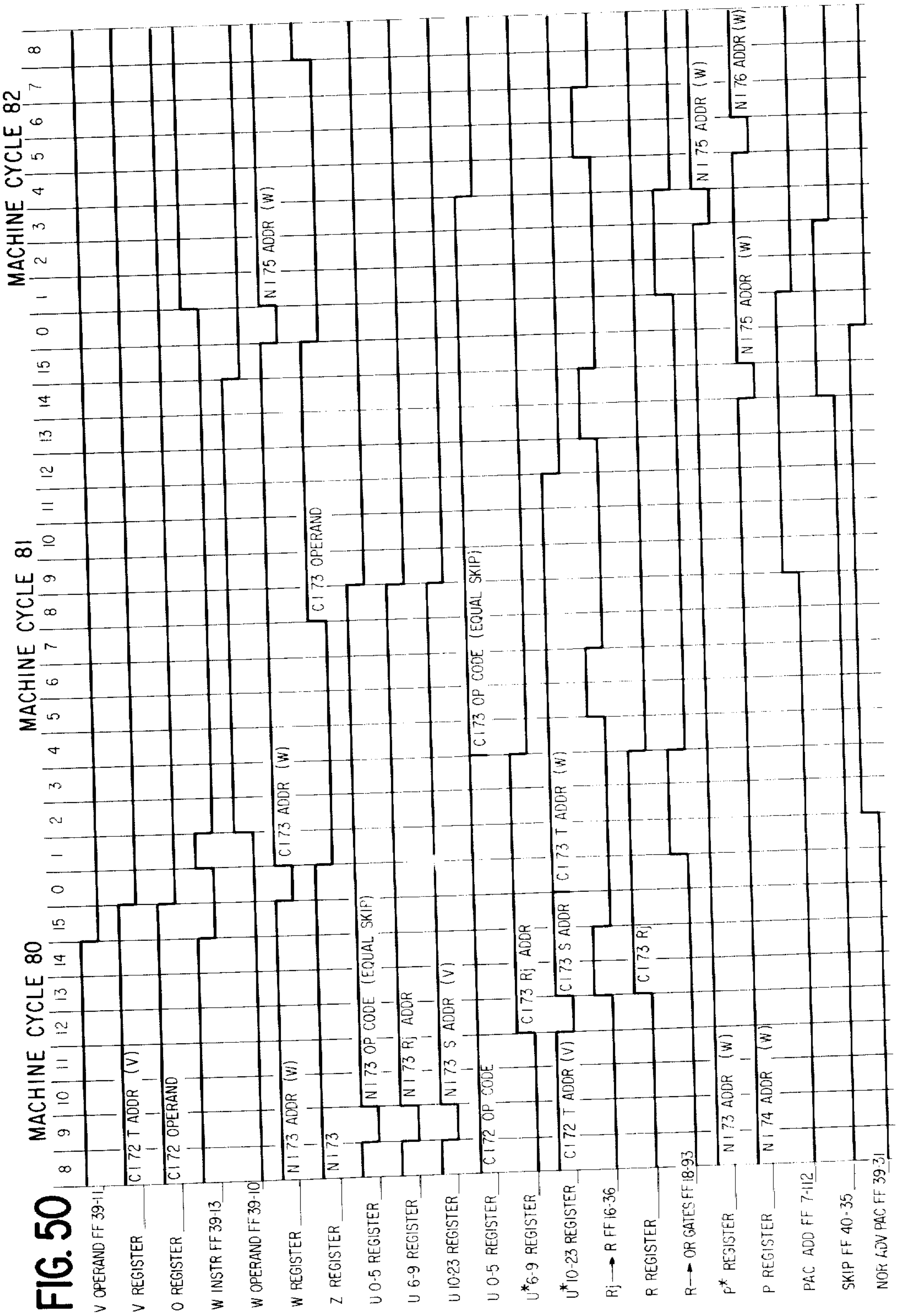
FIG.49

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 82

FIG. 50



STORED PROGRAM SYSTEM

Filed Sept. 13, 1961

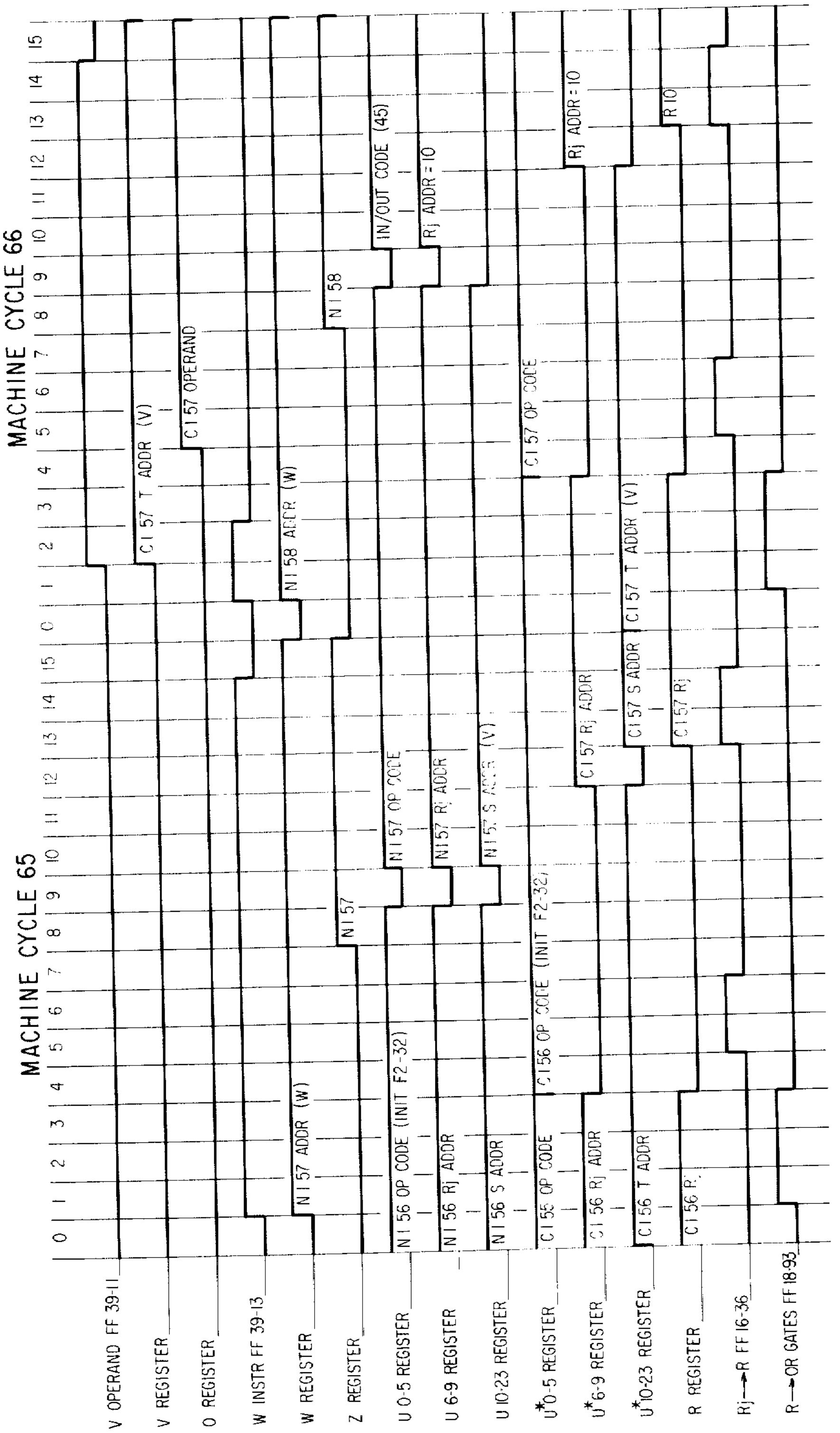


FIG. 51a

STORED PROGRAM SYSTEM

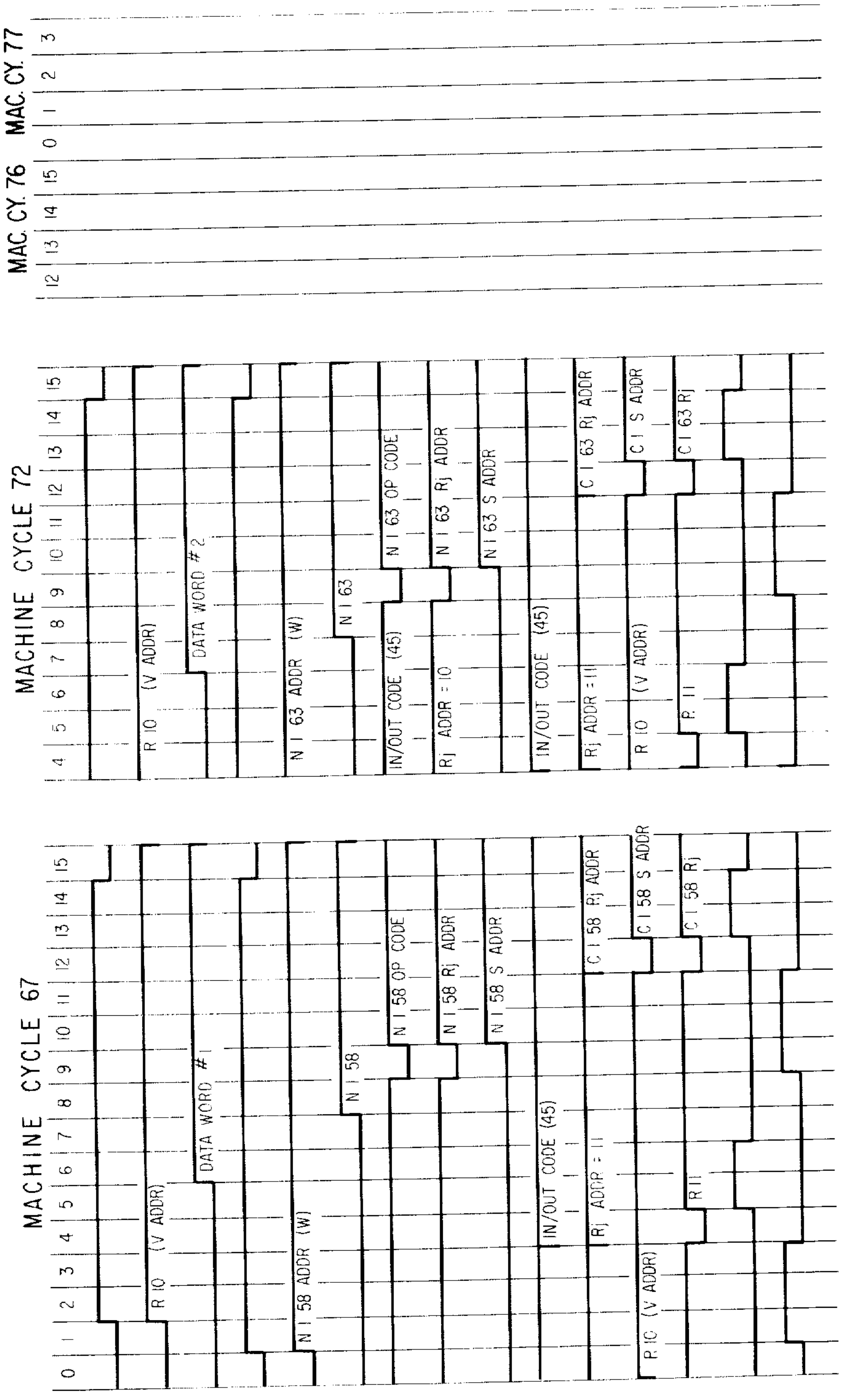


FIG. 51b

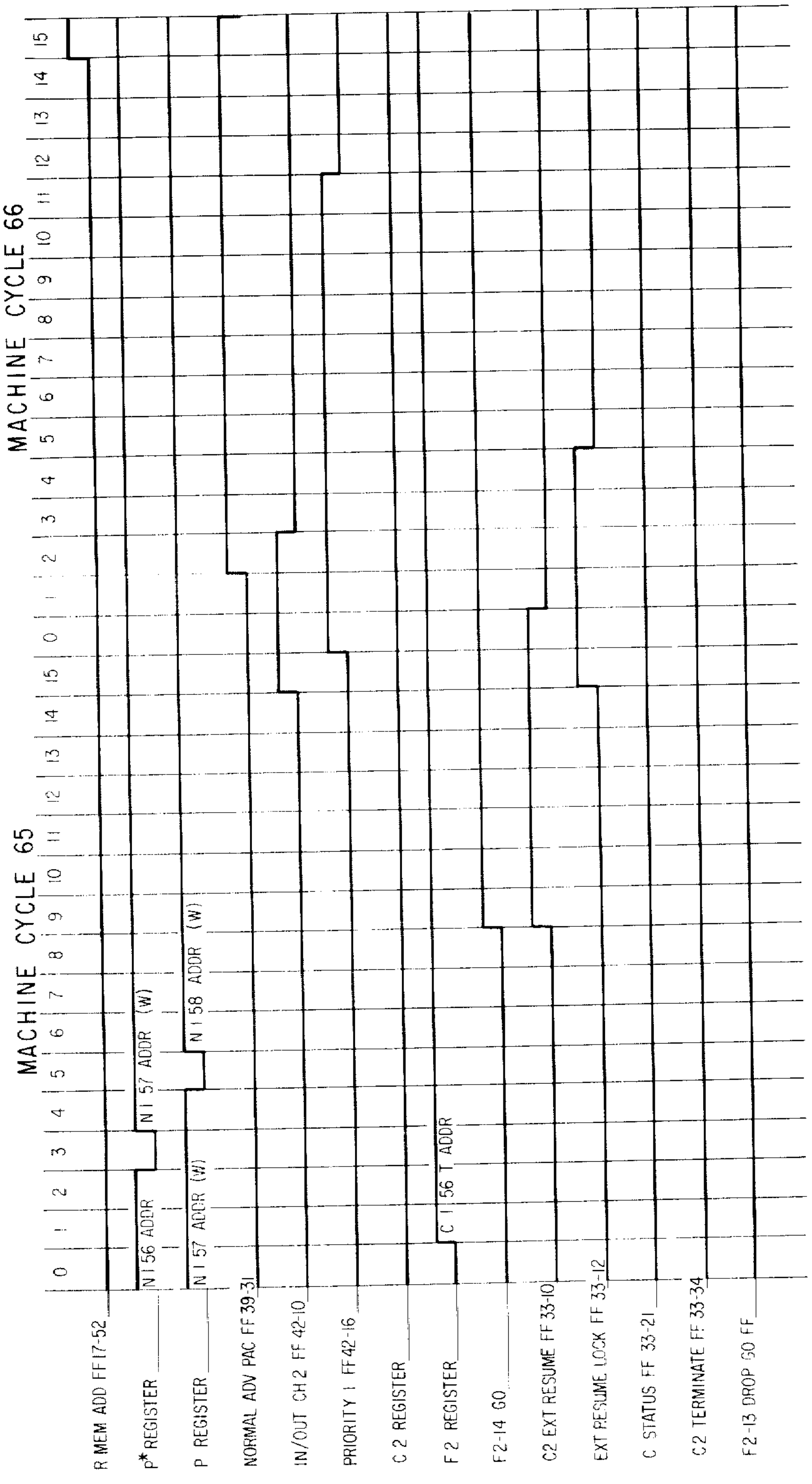

MACHINE CYCLE 65
MACHINE CYCLE 66
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
N I 56 ADDR
N I 57 ADDR (W)
N I 57 ADDR (W)
N I 58 ADDR (W)
C I 56 T ADDR
R MEM ADD FF17-52
P* REGISTER
P REGISTER
NORMAL ADV PAC FF 39-31
IN/OUT CH 2 FF 42-10
PRIORITY 1 FF 42-16
C 2 REGISTER
F 2 REGISTER
F2-14 GO
C2 EXT RESUME FF 33-10
EXT RESUME LOCK FF 33-12
C STATUS FF 33-21
C2 TERMINATE FF 33-34
F2-13 DROP GO FF

FIG. 51c

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 86

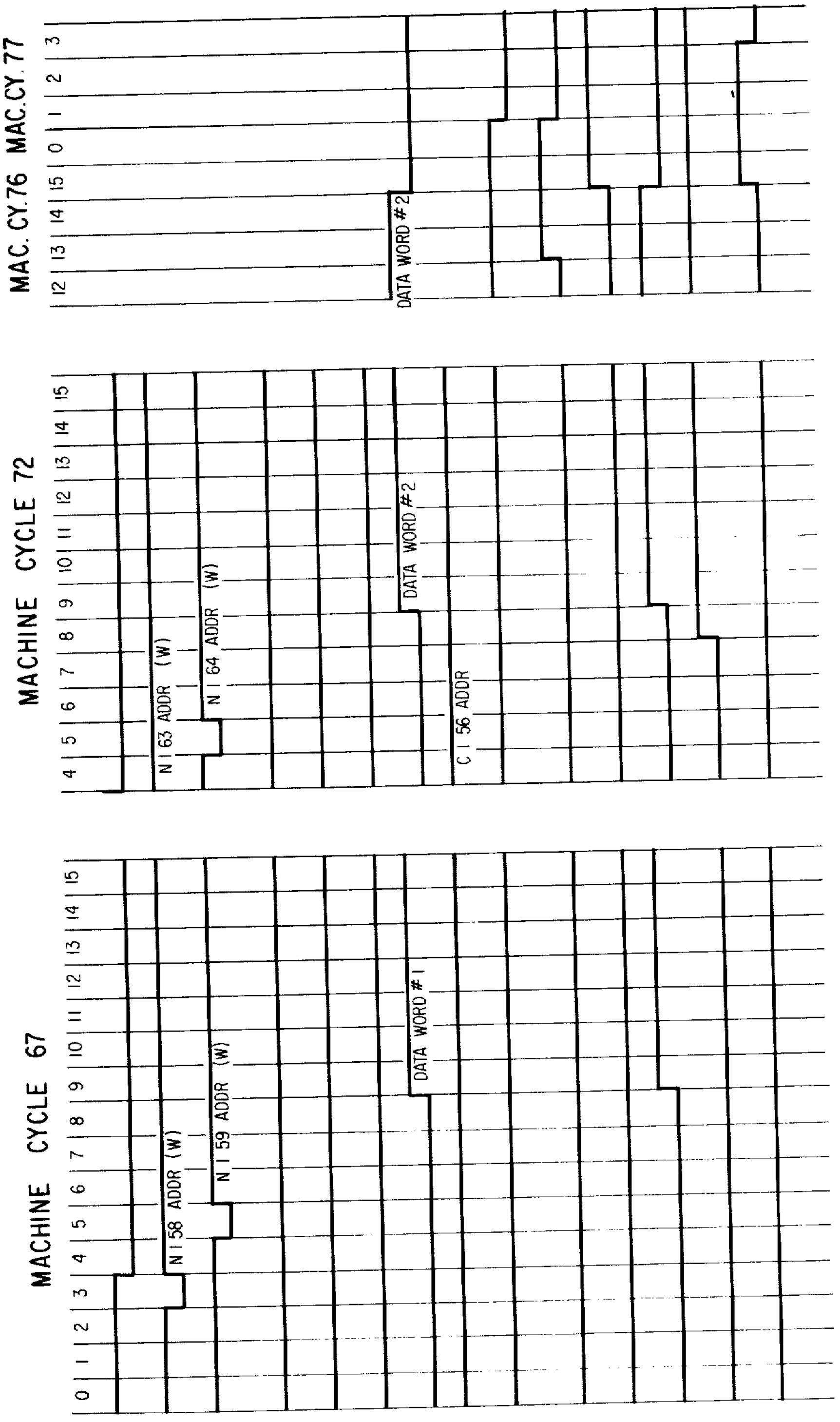


FIG. 51d

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 87

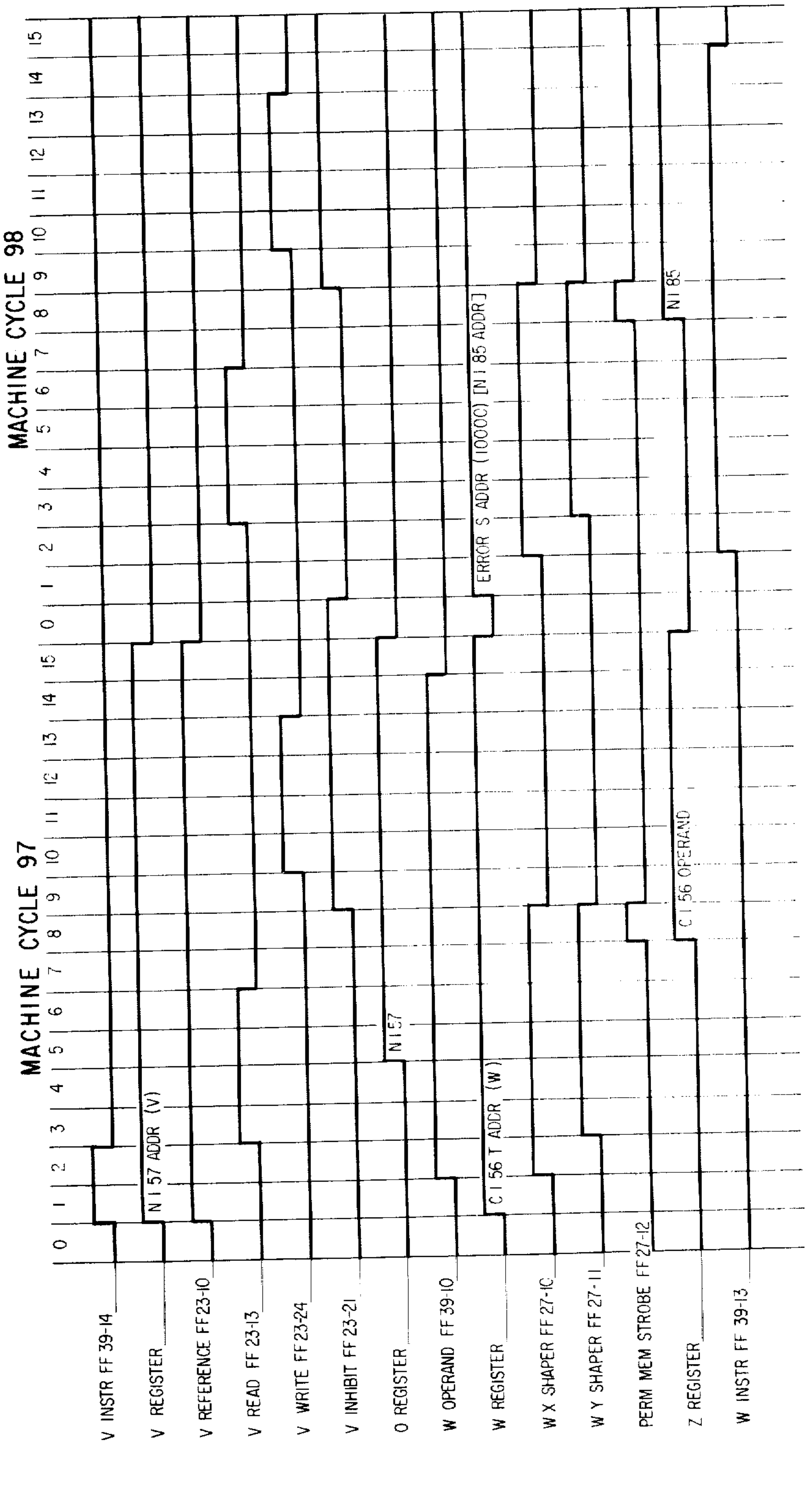


FIG. 52a

STORED PROGRAM SYSTEM

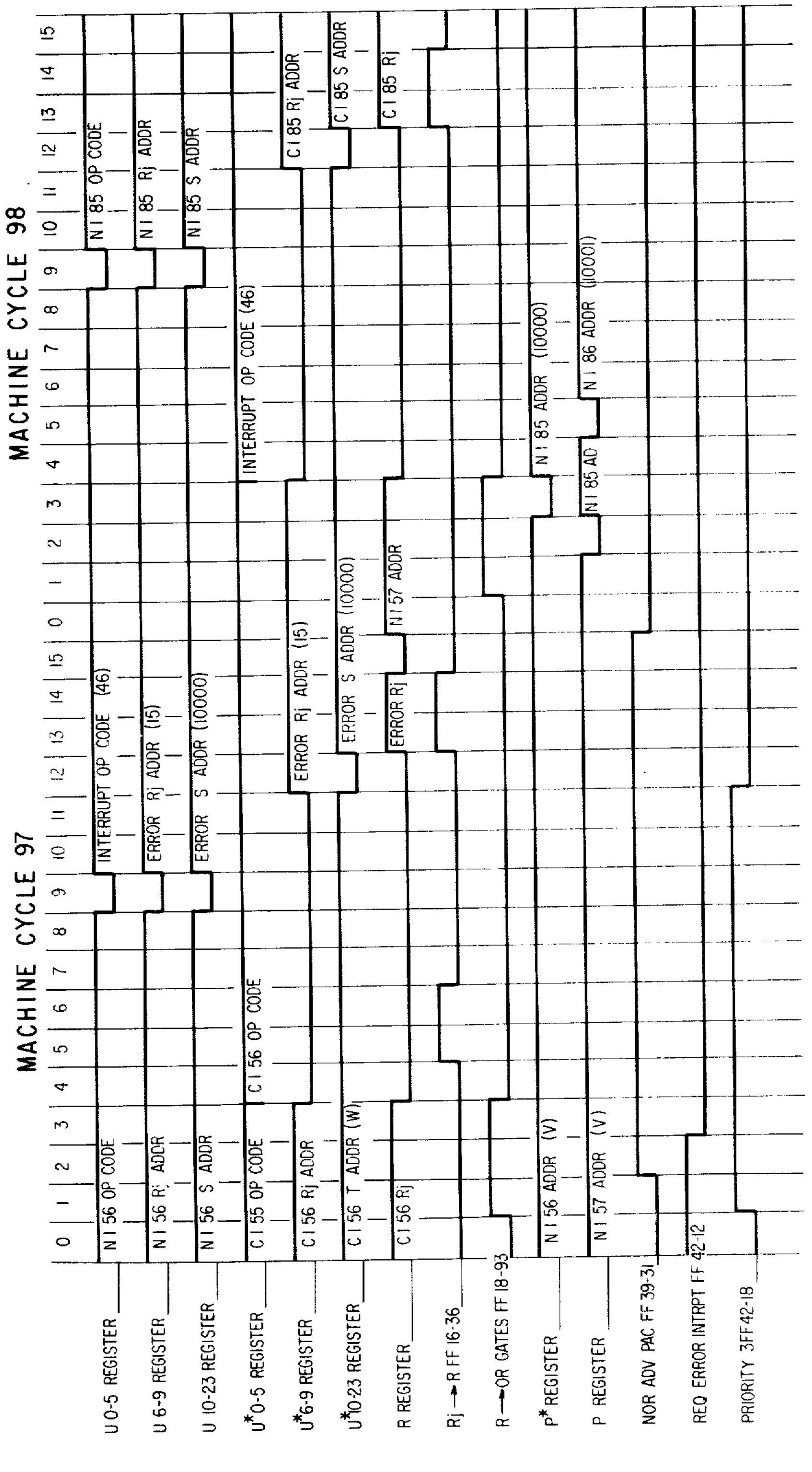


FIG. 52b

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 88

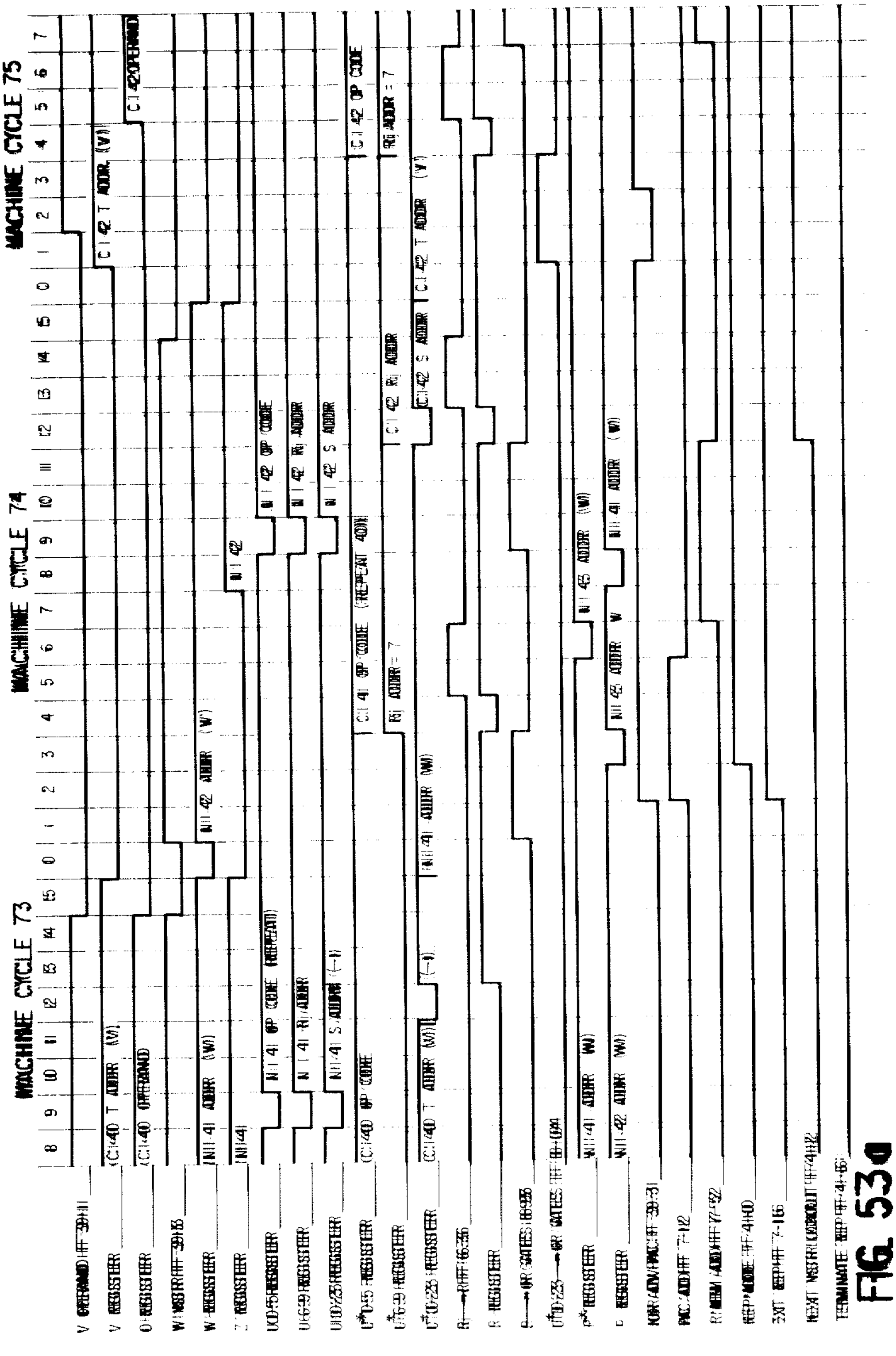


FIG. 53a

July 12, 1966   R. B. ARNDT ETAL   3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961   93 Sheets-Sheet 90

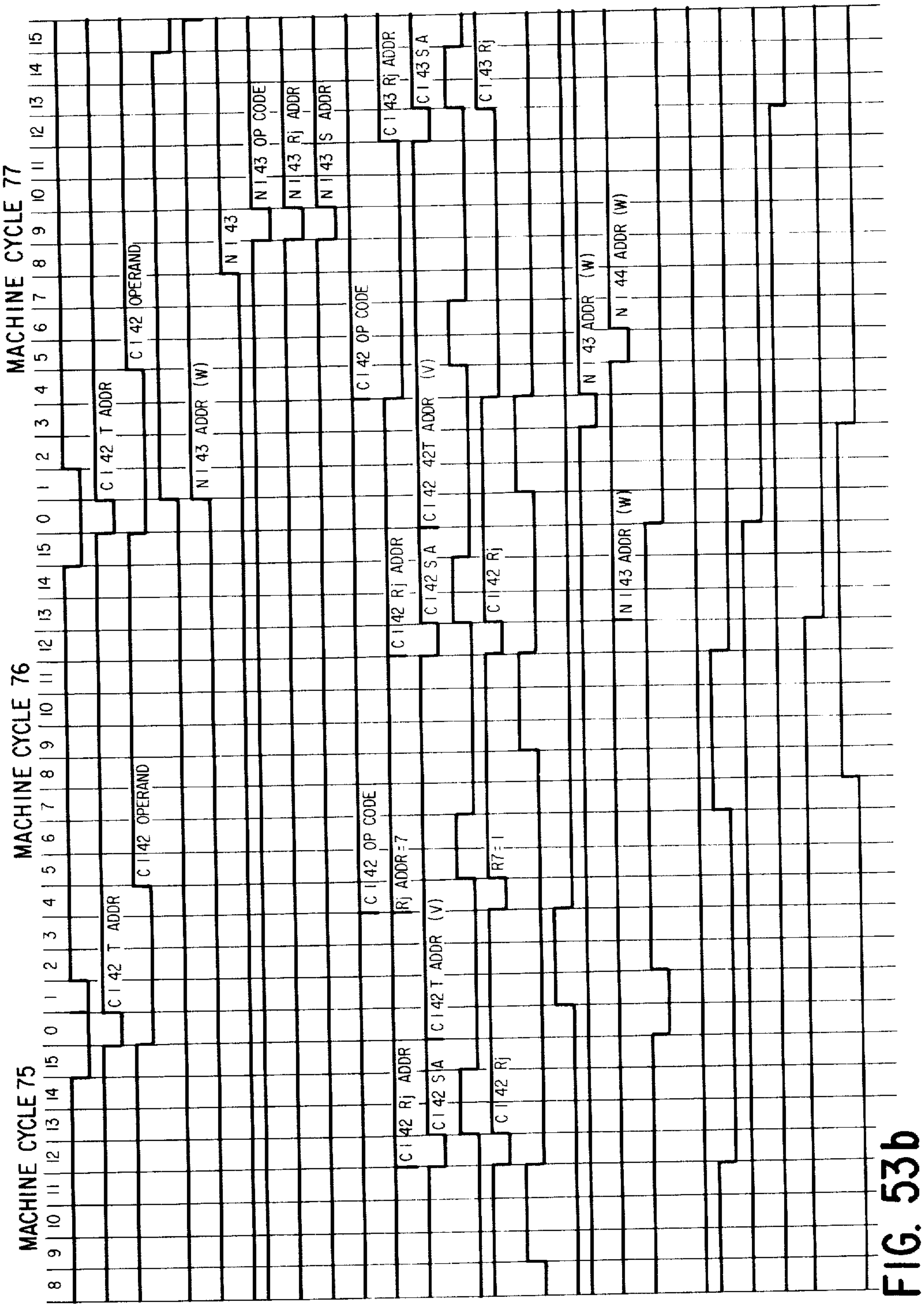


FIG. 53b

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 91

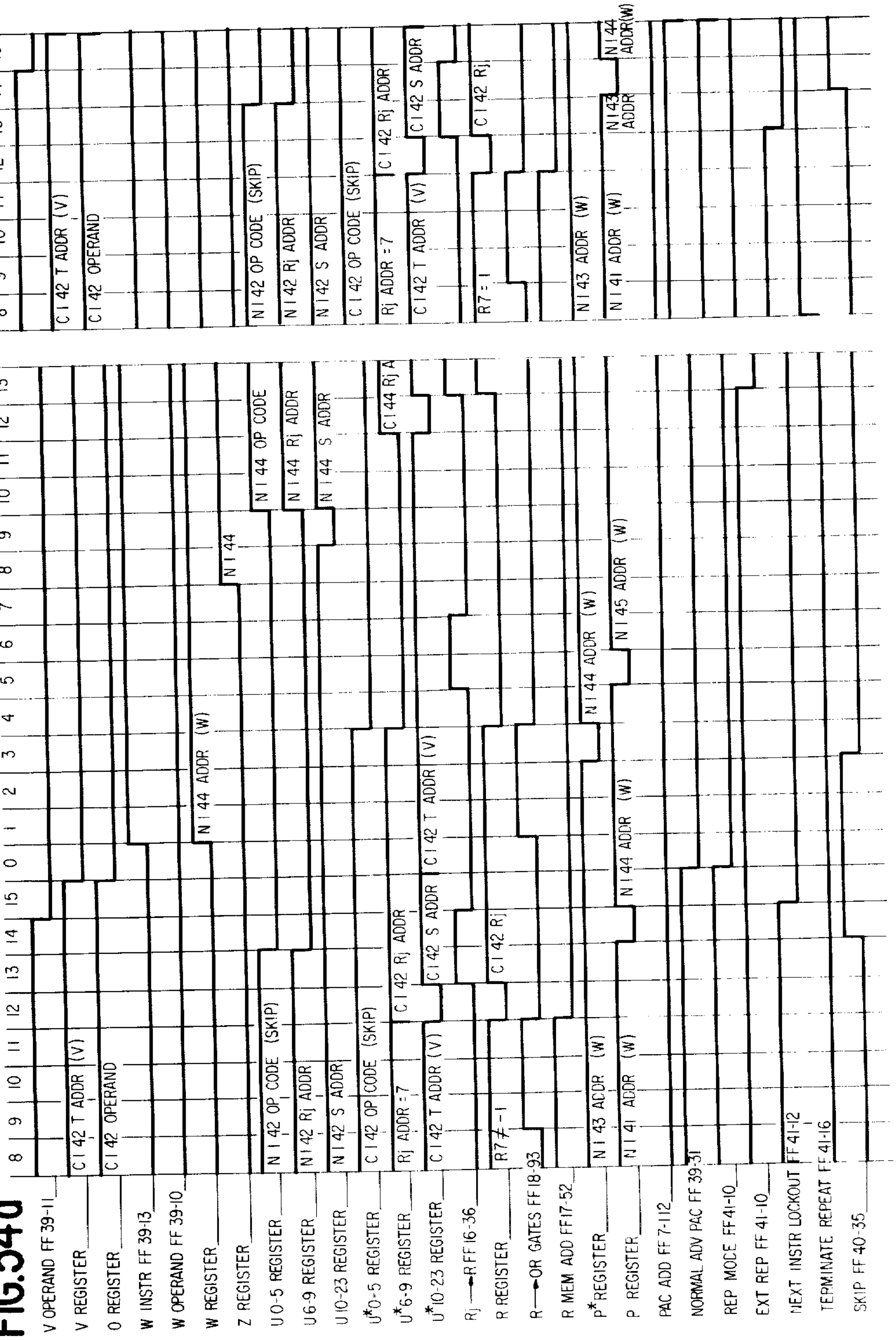


FIG.54a

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 92

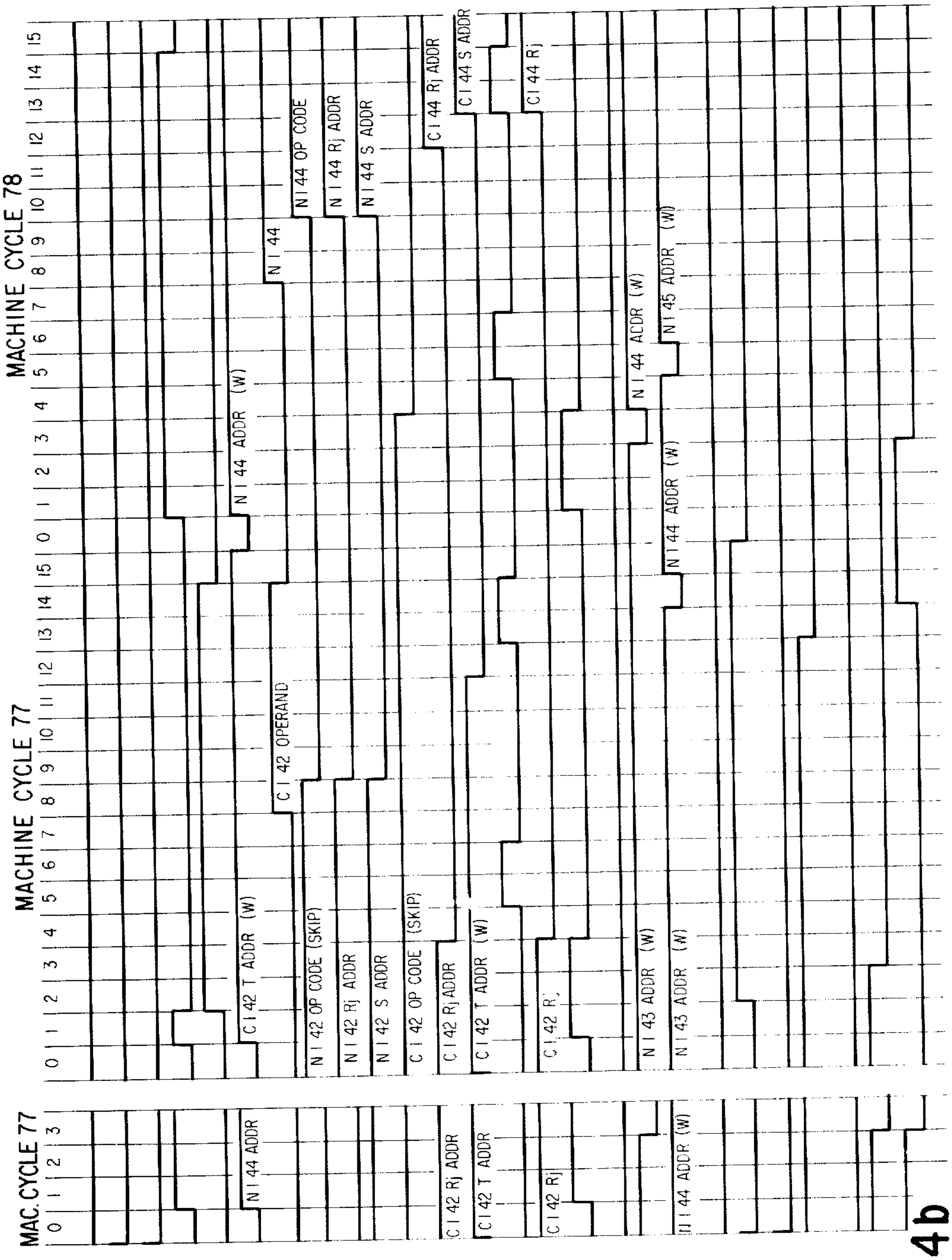


FIG. 54b

July 12, 1966 R. B. ARNDT ETAL 3,260,997

STORED PROGRAM SYSTEM

Filed Sept. 13, 1961 93 Sheets-Sheet 93

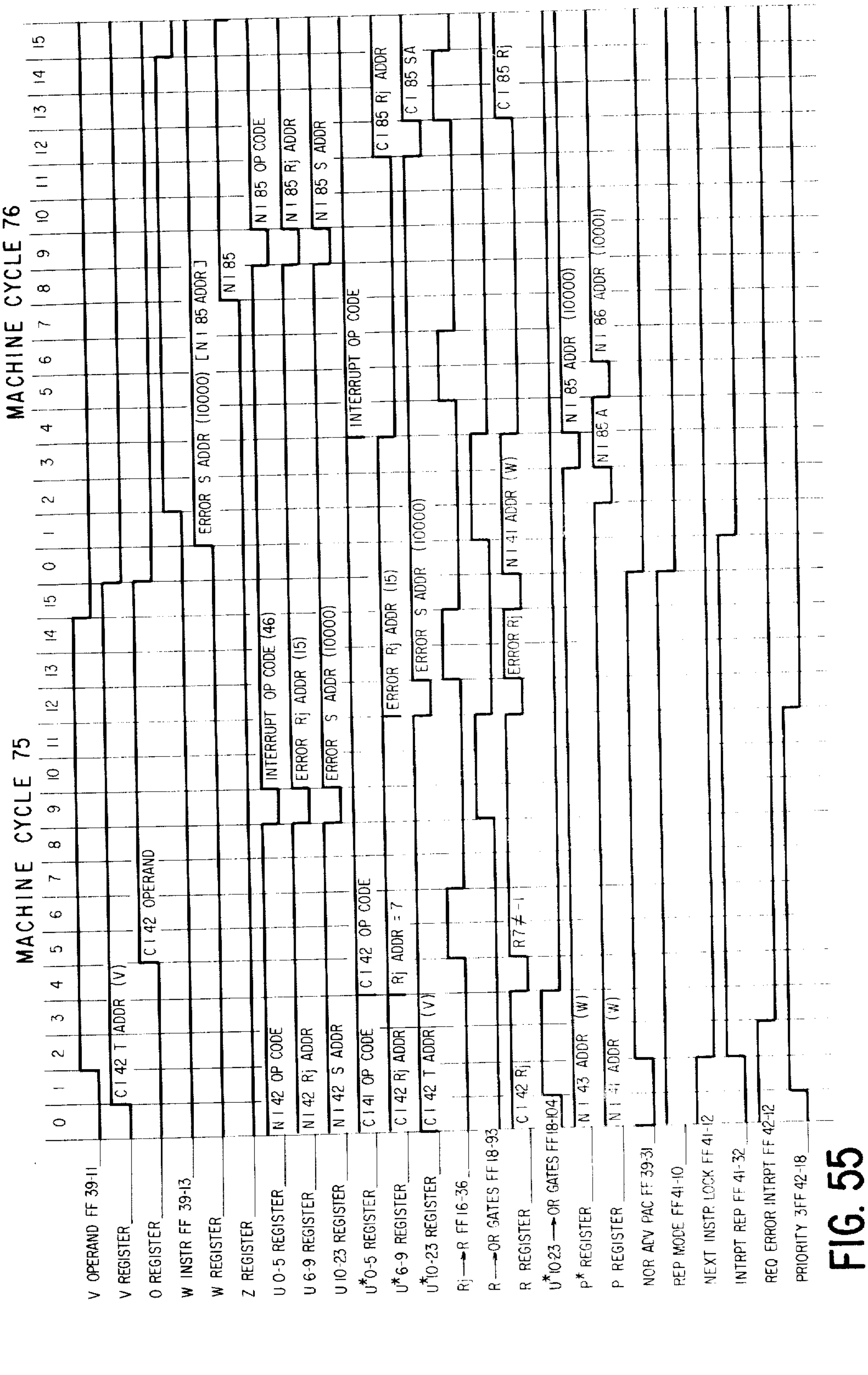


FIG. 55

United States Patent Office

3,260,997
Patented July 12, 1966

3,260,997

STORED PROGRAM SYSTEM

Rolland B. Arndt, Minneapolis, and William Weigler, St. Paul, Minn., assignors to Sperry Rand Corporation, New York, N.Y., a corporation of Delaware

Filed Sept. 13, 1961, Ser. No. 137,795
92 Claims. (Cl. 340—172.5)

This invention relates to instruction sequencing of a stored program data processing system, and more particularly, to one wherein a next instruction can be acquired from memory simultaneously with the execution of a current instruction.

In data processing systems of the digital type, the manipulation and transfer of data words is normally under control of instruction words in a program which is stored internally of the system in a memory unit. The data words are usually also stored within the memory unit. The typical instruction sequencing requires that the memory address of an instruction word be generated to acquire same for subsequently controlling the acquisition and manipulation of a data word within an arithmetic unit or the like. In the so-called single address system, the memory address of each instruction in the stored program is calculated by a unit of the control section designated as a program address counter, or the like. The instruction word itself has a format including at least an operation code specifying the function to be executed, as well as the memory address of a data word to be used in execution of this function. The time required for the processing of an instruction is usually measured commencing from the time that the program address counter references the memory to acquire the instruction itself, and ending with the time that the execution of the instruction using the data word has been terminated. Therefore, for programming purposes, the processing time of each instruction in the program covers a period occupied by two successive memory reference cycles, the first being the reference for the instruction word and the second being the reference for the data word whose address is specified by the previously acquired instruction. Thus, for most programs devised for prior art systems, the number of arithmetic operations which can be processed in a given period of time is substantially less than would be the case if an execution (which includes a memory reference for the operand) could be performed during each machine cycle.

The present invention permits a substantial decrease in effective processing time of instructions of a stored program by normally permitting the acquisition of an instruction word from a memory unit at the same time that the execution of a previously acquired instruction is being performed. Thus, as soon as the execution of a previously acquired instruction is terminated, a new instruction has already been placed into the control unit from the memory unit, so that a new execution cycle can commence immediately without requiring an intervening cycle for instruction acquisition only. Therefore, for purposes of the programmer, each machine cycle of the system can normally be considered one in which a processing of a different instruction will occur. There would therefore be but few machine cycles in which the arithmetic unit of the system cannot be utilized, whereas in the prior art, the arithmetic unit normally stands idle every other cycle. For a given number of instructions in the program, the total time required to process these instructions is therefore substantially less and may approach one-half the time required to process the same number of instructions in the typical prior art system.

The above advantages of the present invention are obtained by providing at least two individual memories each of which can store both instruction and data words, and each of which has its own address and buffer transfer registers for gaining access thereto. The invention also has the ability to reference both memories simultaneously, one for the next instruction and the other for the data word to be used in execution of the previously acquired instruction. This feature is herein designated as an overlap instruction sequencing cycle. However, in certain programs it may be necessary to acquire the next instruction from the same memory that issues the operand data word used in the execution of the current instruction. In such cases, the next instruction does not begin until the operand used in the current instruction is extracted from this memory. Thus, the acquisition of the data word operand for the current instruction takes precedence over acquisition of the next instruction so that the overlap sequencing feature is inhibited, and the next instruction cannot be acquired from its memory until one machine cycle later. In this event, the programmer must consider that the processing time of the next instruction (whose acquisition has been delayed by inhibition of overlap) is extended by a time equal to the duration of one machine cycle.

It is therefore an object of the present invention to provide a data processing system having at least two individual and independent memory units for storing both instruction and data words, together with means for referencing said memory units simultaneously with a view toward extracting an instruction from one and a data word from the other.

Another object of the present invention is to provide a data processing system having at least two independent memory units for storing both instruction and data words wherein acquisition of a data word takes priority over acquisition of an instruction word if both are stored in the same memory unit.

A further object of the present invention is to provide a data processing system of the above described kind which utilizes a double rank instruction register for holding two instruction words at a time.

A further object of the present invention is to provide a data processing system of the above described type which employs single address instructions, and has a double rank program address counter capable of simultaneously holding the memory addresses of two successive instructions of the stored program.

For some instruction words, such as those specifying multiply, divide, square root, and shift operations, the actual execution time in most cases requires more than one machine cycle as compared to execution times of one machine cycle for the simpler arithmetic operations such as add, subtract, etc. The present system provides control circuitry within the arithmetic unit for executing these extended sequence instructions without need for the instruction word to be continuously held in the program unit execution is terminated. Therefore, upon initiation of the extended sequence execution, the program unit may be cleared of the initiating instruction word, and subsequent instrutcion words of the program are acquired from memory and executed by units of the system other than the arithmetic unit. Thus, the present system provides a maximum utilization of time in that the processing of non-arithmetic instructions, which follow an extended sequence instruction, need not be delayed until execution of the extended sequence istruction is terminated. However, if a subsequent instruction is acquired which requires use of the arithmetic unit at the time that the arithmetic unit is executing a previously acquired extended sequence instruction, then the memory instruction sequencing of the system must be inhibited until the arithmetic unit is free to accept another arithmetic instruction.

It is therefore another object of the present invention to provide a data processing system having at least two independent memories for storage of both instruction and data words wherein the acquisition and execution of non-arithmetic instructions can continue with overlap even though a previously acquired extended sequence instruction is being executed by the arithmetic unit.

For certain special instructions, the above described novel overlap feature must be modified. For example, the execution of some instruction words requires that a jump be made in the program sequencing from an instruction memory address held by the program address counter to an instruction memory address held in the instruction word itself. In this case, the program address counter is inhibited from referencing memory for the next instruction during one machine cycle, at which time a memory is referenced by a portion of the instruction word itself. Furthermore, a class of instruction words may require that the execution of an instruction word be prevented, with a skip being made to the next following instruction word in the program. This skip operation is usually conditional upon the result of an arithmetic operation in the arithmetic unit. Therefore, if the skip condition is sensed to be present, the program address counter is required to calculate the address of the next instruction following the instruction to be skipped.

Consequently, yet another object of the present invention is to provide means in the system of the above described type to inhibit the normal operation of a program address counter when a jump instruction is to be executed.

A further object of the present invention is to modify the operation of the program address counter in the system of the above described type if an instruction skip condition is detected.

For repeated executions of a particular instruction word in the program, the technique in the present system is to precede the repeated instruction by a special instruction for the purpose of setting up certain control circuits designed to carry out the repeated operations. In this event, the repeated instruction is maintained in the instruction register until all of the required number of executions is terminated. Therefore, overlap of instruction acquisition and data operand acquisition is inhibited for this period of time so that the repeated instruction cannot be erased from the instruction register by the next following instruction.

It is therefore another object of the present invention to provide control means in the system of the above described type whereby the acquisition of instruction words from memory is inhibited during the execution of a repeated instruction.

Other conditions which affect the overlap instruction sequencing of the present invention are those in which a request is made for the computer to transfer a data word between it and some external unit, and those in which a request is made to interrupt the processing of main program instructions for the purpose of initiating a subroutine opeartion. In either of these cases, the overlap function is inhibited for at least one machine cycle so that special instructions can be forced into the program instruction register from sources other than the individual memories.

It is therefore a further object of the present invention to provide means in a system of the above described type to inhibit overlap, and thereby delay acquisition of an instruction in the program, upon granting of a request for a data word transfer or the initiating of a sub-routine.

These and other objects of the present invention will become apparent during the course of the following description when taken in conjunction with the drawings, which illustrate a preferred embodiment of the invention for carrying out the above described novel features.

FIGURES 1*a* and 1*b* show an overall block diagram of the system incorporating the present invention;

FIGURE 2 is a block diagram of the variable memory access circuits;

FIGURES 3*a* and 3*b* comprise a block diagram of the permanent memory access circuits;

FIGURES 4*a* through 4*f* show the basic logic elements used in the present system;

FIGURES 5*a*, 5*b*, and 5*c* show the P register;

FIGURE 6 shows the P* register;

FIGURES 7*a*, 7*b*, and 7*c* show the PAC Address Counter;

FIGURES 8*a* and 8*b* show U register stages **0** through **5**;

FIGURES 9*a* and 9*b* show the command translator circuits for the U register;

FIGURES 10*a* and 10*b* show U register stages **10** through **23**;

FIGURES 11*a* and 11*b* show U* register stages **0** through **5**;

FIGURES 12*a* and 12*b* show the command translator circuits for the U* register;

FIGURES 13*a*, 13*b* and 13*c* show U* register stages **10** through **23**;

FIGURES 14*a* and 14*b* show U register stages **6** through **9** and U* register stages **6** through **9**;

FIGURE 15 shows the address translator circuits for U* register stages **6** through **9**;

FIGURES 16*a* and 16*b* show the R register;

FIGURES 17*a* and 17*b* show the R Memory Counter;

FIGURES 18*a* and 18*b* show the OR gates;

FIGURES 19*a* and 19*b* show the U* Carry Tree;

FIGURE 20 shows the comparator;

FIGURES 21*a* and 21*b* show the V register and translators;

FIGURES 22*a* and 22*b* show the O register;

FIGURE 23 shows the variable memory control flip-flops;

FIGURES 24*a*, 24*b* and 24*c* show the W register and translators;

FIGURE 25 shows the permanent memory gating logic;

FIGURE 26 shows the Z register;

FIGURE 27 shows the permanent memory control flip-flops;

FIGURES 28*a* and 28*b* show the **F1** register;

FIGURE 29 shows the **C1** register;

FIGURE 30 shows **C1** sequence control;

FIGURES 31*a* and 31*b* show the **F2** register;

FIGURE 32 shows the **C2** register;

FIGURE 33 shows **C2** sequence control;

FIGURE 34*a* and 34*b* show the **D** register;

FIGURES 35*a* and 35*b* show **D** sequence control;

FIGURE 36 shows the **D** counter;

FIGURES 37*a* and 37*b* show the **L** register;

FIGURES 38*a* and 38*b* show the Line Pulse Generator;

FIGURES 39*a* and 39*b* show the memory access control;

FIGURE 40 shows the skip and arithmetic lockout control;

FIGURE 41 shows the repeat control;

FIGURE 42 shows the priority circuits;

FIGURE 43 shows the interrupt and data transfer intruction generator;

FIGURE 44 shows the external real time sync control;

FIGURES 45*a* and 45*b* comprise a timing diagram illustrating operation of the invention when overlap is permissible;

FIGURE 46 is a timing diagram illustrating operation of the invention when overlap is inhibited;

FIGURES 47*a* and 47*b* comprise a timing diagram illustrating operation of the invention during an extended sequence;

FIGURES 48*a* and 48*b* comprise a timing diagram illustrating operation of the invention when executing an instruction jump;

FIGURE 49 is a timing diagram illustrating operation of the invention when executing a selective skip instruction;

FIGURE 50 is a timing diagram illustrating operation of the invention when executing an arithmetic skip instruction;

FIGURES 51*a*, 51*b*, 51*c* and 51*d* comprise a timing diagram illustrating operation of the invention for an input-output data transfer operation;

FIGURES 52*a* and 52*b* comprise a timing diagram illustrating operation of the invention when an interrupt subroutine is requested;

FIGURES 53*a* and 53*b* comprise a timing diagram illustrating operation of the invention when executing a repeated instruction;

FIGURES 54*a* and 54*b* comprise a timing diagram illustrating operation of the invention when repeating the operation of a skip instruction; and

FIGURE 55 is a timing diagram illustrating the operation of the invention when an interrupt subroutine or data transfer is requested during the repeat mode.

In the system disclosed herein, both words of data and words of instruction are stored in memory locations and, when acquired therefrom, are employed in a manner similar to that shown in much of the prior art. For example, an instruction word is comprised of 25 binary bits with parity and includes an operation code portion (OP CODE) specifying the function, e.g., ADD, SUBTRACT, etc., to be executed. A word memory address portion "S" is also included which usually specifies the location of the data word (operand) to be used in execution of the operation code. Alternatively, this address portion can denote the address of the next instruction word to be processed, a constant, a shift control value, or other special values. Instruction words further contain an index address portion "*j*" which may be used to obtain a number from auxiliary storage for the purpose of adding same to the "S" address in order to modify the latter just prior to execution of the operation code. This modified word address portion of the instruction is designated as "T." Data words are also comprised of 25 binary bits with parity and represent numbers in fractional binary notation with a sign bit placed to the left of the binary point. Negative numbers are represented by the 2's complement of positive numbers. Thus, 0.1011 is equal to the decimal value of $\frac{1}{2}+\frac{1}{8}+\frac{1}{16}$ or $[+\frac{11}{16}]$, while 1.0101 is equal to $-1+\frac{1}{4}+\frac{1}{16}$ or $[-\frac{11}{16}]$. The instruction and data word formats are illustrated below.

INSTRUCTION WORD FORMAT

OP CODE						j				S, then T														
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24

Parity (odd)

In general, "S" and "T" specify memory addresses wherein are stored operands. "*j*" specifies an address in an auxiliary memory (here designated as R memory) wherein is stored a modifying index number specified as R_j. Thus, R_j+"S"="T."

DATA WORD FORMAT

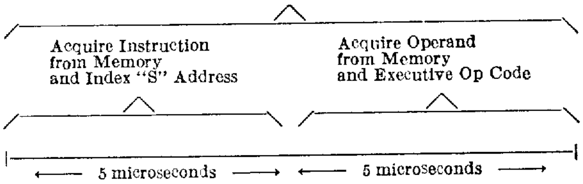
Sign: 0=plus; 1=negative

FRACTIONAL NUMBER																								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24

Binary Point Parity (odd)

Basically, acquisition and execution of a single instruction requires 10 microseconds in the present system. As shown below, the basic instruction processing time is considered in terms of two 5-microsecond periods, where a 5-microsecond period is defined to be one machine cycle.

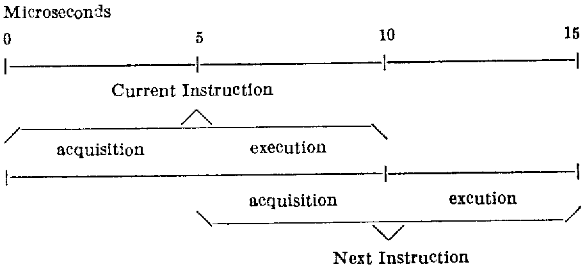
INSTRUCTION PROCESSING TIME


Acquire Instruction from Memory and Index "S" Address
Acquire Operand from Memory and Executive Op Code
5 microseconds
5 microseconds

TIME RELATIONSHIP IN THE BASIC INSTRUCTION

Common Commands, used to acquire an instruction, are executed during the first 5-microsecond period. During the next 5-microsecond period, the operand is acquired, and then used in the execution of the instruction.

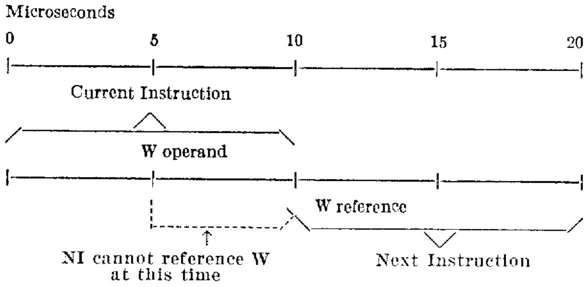
In order to effect a maximum utilization of time, the present invention is designed logically in a way that permits the simultaneous processing of certain portions of successive instructions. Thus, the memory acquisition time associated with the current instruction occurs during the same actual time as the execution time associated with the previous instruction. Likewise, the execution time associated with the current instruction and the memory acquisition time of the next instruction occur simultaneously. This association of successive instructions is illustrated below.



NORMAL ASSOCIATION OF SUCCESSIVE INSTRUCTIONS

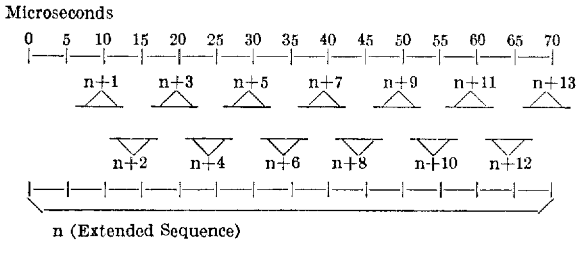
The illustration above shows that the computer may handle a new instruction every five microseconds. Therefore, for programming purposes, five microseconds can be considered the basic processing time requirement for any given instruction.

Two individual memories designated as W and V, plus the ability of the computer to reference both memories simultaneously, make this instruction-overlapping feature possible. However, in certain programs it may be necessary to acquire the next instruction from the same memory that issues the operand used in the current execution of the instruction. In such cases, the processing of the next instruction does not begin until the operand used in the current instruction is extracted from memory, thus adding five microseconds to the program at this point as shown below.



THE ASSOCIATION OF SUCCESSIVE INSTRUCTIONS WHEN AN ATTEMPT IS MADE TO SIMULTANEOUSLY REFERENCE THE SAME MEMORY

During the Multiply, Divide Square Root, and Shift instructions (hereafter called extended sequences), special conditions exist regarding the overlap of instructions. For these instructions from two to thirteen machine cycles may be required for execution by the arithmetic unit. Only instructions which are non-arithmetic may be processed during the existence of an extended sequence since the arithmetic unit can only operate on one instruction at a time. During the extended sequence, each non-arithmetic instruction is handled in the conventional manner. A normal succession of non-arithmetic instructions during an extended sequence is subject to the same conditions as a normal succession of arithmetic or non-arithmetic instructions during those times that an extended sequence does not exist. A maximum of 13 non-arithmetic instructions can be processed during the extended sequence as shown below. The only instruction shown (other than the extended sequence itself) that may be arithmetic is $n+13$.



THE RELATIONSHIP OF A SERIES OF NON-ARITHMETIC INSTRUCTIONS AND AN EXTENDED SEQUENCE

The number of instructions that is processed during an extended sequence reduces proportionally as the number of special conditions increases. Other conditions that influence the processing times of instructions are those in which a jump or a skip instruction is being executed, or when an instruction is being repeated. Furthermore, the request by the system of an input/output data transfer or for an interrupt sub-routine also affects the normal program sequence.

TABLE OF CONTENTS Columns

OVERALL SYSTEM

FIGURE 1 is a block diagram of the overall system, which will first be described in order to subsequently make clear the relationships between the individual detailed units. The transmission of data between functional units is depicted with heavy connecting lines. Control paths between functional units are indicated by light connecting lines. The arrowhead(s) indicates in what direction data may be transferred from one functional unit to another or the relative paths of control between functional units. Arrowheads on both ends of a line indicate that data transmission or control paths may occur in either direction. A circle with a number within it is placed on all data flow lines. The number indicates the number of bits that are contained in the binary quantity that can be transmitted. Whenever a dot also appears within the circle, this indicates that the data is "double gated" from one functional unit to another. Double gated should be construed to mean that both the "0's" and "1's" of a binary quantity are transmitted. If no dot appears within the circle, the quantity is single gated, which means that only "1's" are transmitted. The letter (s) "A," "C," "I/O," "M," or "E" next to a particular block indicates that it is a part of the arithmetic, control, input/output, memory, or error units respectively.

1.0. *Control*

The Control Section receives, interprets, and directs the processing of each instruction. In directing the execution of an instruction, the Control Section exercises supervision over the activities of the computer by controlling the timing of all computer operations. The principal circuits of the Control Section are the Interrupt and Data Transfer Control, Instruction Register (U and U* registers). Command Translator Circuits, Subcommand and Memory Access Control, Main Pulse Distributor, Program Address Counter (P and P*), and R Memory Control. During the execution of an instruction, the instruction word is held in the Instruction Register. The Command Translator Circuits interpret the operation code portion of the instruction and determine which commands are required for its execution. The Subcommand and Memory Access Control interprets the address portion of the instruction and determines which memory section is to be referenced. The Command Translator Circuits and the Subcommand and Memory Access Control generate the commands in a specific sequence of steps necessary to execute the instruction. The Interrupt and Data Transfer Control enables the computer operation to be interrupted and directed to perform special operations. The R Memory Control is normally employed to reference R Memory and withdraw a number used to index the "S" address portion of the instruction word to form the "T" address portion.

Program Address Counter.—The Program Address Counter provides the memory address of the next instruction to be acquired. The memory address is entered in the Program Address Counter either manually or automatically from the Instruction Register as the result of an Interrupt or Jump instruction. As each instruction is acquired, the Program Address Counter normally advances by one count and thus presents the next address in order. In specific cases, however, the current instruction being executed will prevent the normal advancing of the Program Address Counter. For example, a skip instruction forces the Program Address Counter to advance by two, while a Jump instruction or an Interrupt will force a new address into the Program Address Counter. An address is generated by the Program Address Counter in the following manner:

P* is cleared
P is transmitted to P*
P is cleared
(P^*+n) is transmitted to P (where $n=1$ or $n=2$)

P Register.—The P register is a 14-bit register which holds the memory address of the next instruction after it is computed by the Address Counter. An address can also be placed in P from U*10–23 if a Jump instruction is being performed. The P register communicates with the W and V address registers and with the R transfer register.

Address Counter.—The Address Counter provides a means for adding either one or two to the quantity in P*, i.e., it computes the address of the next instruction.

P Register.*—The P* register is a 14-bit register which holds the address of the instruction currently being performed.

Instruction Register.—This provides enabling signals to the Command Translator Circuit, R Memory Read and Word-Write Circuits, and the Subcommand and Memory Access Control. The address portion of instruction is transmitted to one of several possible destinations as determined by the Command Translator Circuits and Subcommand and Memory Access Control. For example, if the instruction code specifies a store operation, then the address portion will be treated as an address and transmitted to the appropriate memory as directed by the Subcommand and Memory Access Control. If the instruction is interpreted as a selective sense instruction, then the address poriton of the instruction will be treated as an operand to determine whether or not the action requested will occur. (The selective sense instructions are always sensed and interpreted, but are only executed if a corresponding sense selection has been made at the Computer Control Panel.)

In general, the Instruction Register receives an instruction word from either the Variable or Permanent Memory (or manually from the Computer Control Panel), translates the operation code into commands, modifies the Variable or Permanent Memory address by some quantity from R Memory if necessary, and uses this modified or unmodified Variable or Permanent Memory address to obtain an operand from storage if one is required for the instruction. The commands issued by the Command Translator Circuits make possible various operations by different sections of the computer so that the instruction can be performed in logical, sequential steps.

The registers which comprise the Instruction Register and the main data paths for data transmitted to and from the Instruction Register are shown in FIGURE 1.

U and U Registers.*—These registers are 24-bit registers used to hold a complete instruction word while various operations necessary for the execution of the instruction are being performed. A quantity can be entered into **U0–23** via the Z or Q transfer register, or manually from the Computer Control Panel (not shown).

(a) **U0–5**: This portion of the U register holds the

operation code of an instruction while the code is being translated by the Command Translator Circuits.

(b) U***0–5**: This portion of the U* register holds the operation code (after its transmission from **U0–5**) for further translation of the instruction by the Command Translator Circuits.

(c) **U6–9**: This portion of the U register holds the address of an R Memory index word which is referenced by the "*j*" address portion of the instruction. If no R memory word is required, no quantity is placed in **U6–9.**

(d) U***6–9**: This portion of the U* register holds the *j* address (after its transmission from **U6–9**) of an R Memory word also. From this register the address can be translated, and the R Memory address specified can be referenced.

(e) **U10–23**: This portion of the U register holds the "S" address of an operand in Permanent or Variable Memory or the Accumulator. The operand is required for the execution of the instruction.

(f) U* Add Tree: The U* Add Tree makes possible the addition of some quantity in the R register (R Memory) to the quantity in **U10–23**. This operation is especially useful for indexing. For some instructions, **U10–23** is modified by the addition of the contents of the R register, and for others **U10–23** is left unmodified and placed directly into U***10–23.**

(g) U***10–23.** This portion of the U* register holds the modified "T" (or unmodified) Variable or Permanent Memory or Accumulator address of the instruction operand. Register bits U***10–23** are the main portion of the Instruction Register that communicate directly with various sections of the computer.

Command Translator Circuits.—The Command Translator Circuits sense the output of the six highest order bits of the Instruction Register (**U0–5** and U***0–5**), and translates each of 35 possible operation codes into a unique set of command enables. Timing pulses, generated by the Main Pulse Distributor, strobe these enables and thereby generate the commands in the sequence necessary to carry out the steps in the execution of each instruction.

Subcommand and Memory Access Control.—The Subcommand and Memory Access Control directs the referencing of all memory sections except R memory. It senses the outputs of the Instruction Register, Program Address Counter and the Command Translator Circuit and determines which memory section is to be referenced. Using timing signals generated by the Main Pulse Distributor, the Subcommand and Memory Access Control issues the commands to reference the specific memory section.

There are two primary functions of Subcommand and Memory Access Control. The first primary function is the control of the computer during the execution of a Repeat, Selective Sense or Skip instruction, or during the execution of an "extended sequence instruction" (arithmetic lockout). An extended sequence instruction is an instruction that requires more than 5 microseconds (one machine cycle) for its execution; e.g., Divide, Multiply, etc. The second primary function is that of Permanent or Variable Memory access (Memory Access Control). The units comprising the Subcommand and Access Control are the following.

Repeat Control.—Repeat Control governs the repeat of an instruction and provides the circuitry necessary for the correct interruption of a Repeat instruction. If a Repeat instruction is interrupted, it is completed after the interrupte has been satisfied.

Skip and Arithmetic Lockout Control.—Skip and Arithmetic Lockout Control governs the performance of a Selective Sense Or Skip instruction and also permits the computer to delay the execution of an instruction following an extended sequence instruction until the extended sequence is completed, if the next instruction is an arithmetic instruction (requires use of the Arithmetic Section). All extended sequence instructions are performed in the Arithmetic Section.

Memory Access Control.—Memory Access Control governs the access to either Permanent or Variable Memory, or to the Accumulator. Instructions and operands can be referenced in memory. An instruction is a computer word that initiates a particular operation, e.g., Add, Load A, etc. An operand is a quantity (constant) referenced by the address portion of the instruction word that is to be used in the performance of the instruction.

In summary, Subcommand and Memory Access Control govern the special operations of the Repeat, Selective Sense, Skip, and extended sequence instructions. It also controls the access or referencing of a word in Variable or Permanent Memory or the Accumulator.

Interrupt and Data Transfer Control.—The Interrupt Control circuits provide for the acceptance and processing of interrupt signals from both external and internal units. Interrupt signals are received by the Interrupt and Data Transfer Control and are assigned a priority. An interrupt signal is a command to the computer to discontinue present operation and process a new instruction. An instruction is generated by the Interrupt Control circuits and is forced into the Instruction register which then executes the instruction. All operations affecting the starting of the computer or deviating from a program must be processed by the Interrupt and Data Transfer Control.

Interrupt and Data Transfer Control consists of three basic parts: External Real Time Sync Control, Priority Circuits, and the Interrupt and Data Transfer Instruction Generation circuits. These circuits act collectively either to generate interrupts internally or to produce Interrupt and I/O Data Transfer instruction upon request from various sections of the computer. Any program being performed by the computer may be interrupted by an Interrupt instruction. If an interrupt request is received, all other interrupt requests are locked out for six instruction times so that an interrupt program can be performed. At the end of 6 instruction times, another interrupt program can be performed if an interrupt request is held in the Priority Circuits; otherwise the interrupted program will be resumed. Data transfers can be performed regardless of interrupt priorities. Also, no six-instruction-time lockout exists for I/O Data Transfer instructions because they do not disrupt the main program completely as Interrupt instructions do.

External Real Time Sync Control.—External Real Time Sync Control produces Start Pulse and CI interrupt requests when a Start Pulse or CI interrupt request is received externally from a critical external device. External Real Time Sync Control also gates the Error Interrupt request to the Priority Circuits.

Priority Circuits.—The Priority Circuits assign interrupt and data transfer requests, prearranged priorities, and also generate the 6 instruction-time lockout required by an interrupt program.

Interrupt and Data Transfer Instruction Generation. This portion of Interrupt and Data Transfer Control produces automatically the proper operation code, "*j*" reference, and memory address comprising a particular Interrupt or I/O Data Transfer instruction.

In summary, Interrupt and Data Transfer Control receives the interrupt and data transfer requests from external sources. If necessary, these interrupt and data transfer requests are held, and at the proper time the first instruction of their respective program (for interrupts) is placed in **U0–23**. I/O Data Transfer instructions do not call for the execution of special programs. Therefore, a data transfer operation requires only that the Data Transfer instruction be place in **U0–23**.

Main Pulse Distributor.—The Main Pulse Distributor produces the main pulses upon which all computer operations are based. The distributor produces a cycle of 16 main pulses, 0 through 15, then repeats this cycle. Main pulses produced by the Main Pulse Distributor are pulse signals that are produced in a numerical sequence, i.e., 0, 1, 2, 3, through 15. There are 16 main pulses in a main pulse cycle. Each main pulse cycle is 5 microseconds in length. These main pulses are identified by the numbers 0 through 15 written MP0, MP1, etc.). Main pulses are issued to all sections of the computer to enable the performance of instructions in a logical, time-ordered sequence. This unit is not shown in detail in the instant application.

R Memory Access and Control.—R Memory Access and Control provides access to the R memory. It also provides for the comparison of U***10–23** with R register and for the addition of 1 to the quantity in the R register.

R Register.—The R register is a 14-bit data transfer register which provides an output communication path between R Memory and other sections of the computer. It contains, among other values, the number used to index the "S" address portion of an instruction.

OR Gates.—The OR Gates provide a 14-bit path for writing a quantity into R Memory from several sources, e.g., U***10–23**, X register, etc.

R Memory Counter.—The R Memory Counter provides the necessary circuitry for adding 1 to the R register if required and then entering (R+1) into R Memory via the OR Gates. Also, the content of R register may be gated through the counter without any change.

Comparator.—The Comparator compares R with U***10–23** to determine when a data transfer operation should be terminated.

In summary, words which are read out of R Memory enter the R register and are thereafter distributed to the various portions of the computer. Words which are to be written into R Memory are transmitted to R Memory via the OR Gates. The quantity (R+1) can be written into R Memory via the OR Gates if required.

1.1. *Memory*

The Memory Section of the system is made of three independent units of storage: Reference memory (designated R Memory), which consists of a small capacity, high-speed, destructive readout (DRO) ferrite core storage unit; Variable Memory, which consists of a medium capacity, destructive readout storage unit using a conventional magnetic core storage system; and Permanent Memory, which consists of a large capacity nondestructive readout (NDRO) storage unit using permanent magnets and a twistor matrix.

A fourth unit, the Twistor Input Switch (Twistor Input Unit) is associated with permanent Memory and acts as a high-speed input device between certain critical external units and the computer.

The characteristics of the three memory units are high-speed operation, random access to stored data, and retention of data when power is turned off (nonvolatility). All three storage units have provisions for checking parity. Variable Memory, Permanent Memory, and the Twistor Input Switch each provide for storing parity along with the data word. Parity for the R Memory is stored in the Error Circuits.

R Memory.—The R Memory is recognized as a part of the Control Section because of the tasks it performs in modifying the address portion of instruction words and in storing data transfer addresses during subroutines. The description presented in the Memory Section treats only the physical operation of the unit as an independent storage device. For an explanation of R Memory as an integrated part of the complete system, refer to the above description of the Control Section. The R Memory is a magnetic core storage device capable of storing fifteen 14-bit words at address locations **1** through **15**. The memory is word-organized and employs a coincident current write and a single current read cycle. Four bits of the instruction word are required to specify an address in R Memory. Bits 6, 7, 8 and 9 of U*, the second rank of the instruction register, are translated to select one of 15 word addresses. Information is inserted into R Memory through the OR gates. This information may come from the Q, X, U* or R registers. R, the 14-bit rest of the computer and back into R Memory if desired. Parity for R Memory is stored and checked in the Error Circuits.

The Read and Word-Write Circuits provide the drive current for switching the state of selected memory devices. One Read Circuit and one Word-Write Circuit is associated with a specific address. Their selection is determined by enable signals from the translator circuits of bits 6–9 of the Instruction Register U*. A Read Circuit, when enabled by U***6–9**, applies a full drive signal to the associated memory devices which drives them all to the "0" state. The Word-Write Circuits, when enabled, apply a half write signal to the associated memory devices, which is not by itself of sufficient value to drive them to the "1" state. The additional force necessary to drive a given element to the "1" state is supplied by the Digit Write Circuits. Therefore, only those elements which receive the Digit Write signal in addition to the Word-Write signal will switch to the "1" state.

The R-Memory device store the information. The element employed is the magnetic core. The cores are arranged in rows and columns in a 16 x 16 matrix. One Read wire and one Word-Write wire is threaded through each row of cores; one Digit Write wire and one Sense Wire is threaded through each column of cores. The application of signals to these wires enable the cores to switch to either the "0" or "1" state. The Sense Amplifiers receive and amplify the sense signals during a Read operation. The outputs set appropriate stages of the R-Register. The Digit Write Circuits supply drive current necessary to complement the Word-Write signal to effect the storing of "1's" in R Memory. Enables are supplied from the OR Gates.

Variable Memory.—The Variable Memory is a conventional magnetic core storage system. The unit contains 2048 address word locations capable of storing a 24-bit instruction word or operand (data word) plus parity. Characteristics are high-speed operation, random access to stored data, and retention of data when power is turned off (nonvolatility). The time required for one memory reference (the basic memory cycle) is 5 microseconds. Variable Memory is used to store input-output data, the results of intermediate calculations, error addresses, and variables. Except during the Real Time Mode, the Variable Memory may also be used to store instruction. A block diagram of Variable Memory and associated control circuitry is shown in FIGURE 2.

A read operation followed by a write operation make up what is called the basic cycle in Variable Memory. The duration and timing of these operations is controlled by the Variable Memory control flip-flops and clock pulses from the Main Pulse Distributor. Major sequence commands for the transmission of address information to the V register and for data transmission to and from the Q register are issued by the Control Section.

V, the 11-bit address register, selects one of the 2048 address locations during a Variable Memory reference. The translators associated with V simultaneously enable Xand Y access switches (selectors) and drivers at opposite ends of the drive lines to permit the flow of current through the cores. Once an address has been selected, data is transferred from Variable Memory to the rest of the computer and from the rest of the computer to Variable Memory through Q, the 24-bit transfer register.

In addition to transferring a 24-bit data word, the Q register also contains an additional bit for storing parity. Parity is generated when data is sent to Q for storage in Variable Memory. The parity thus generated is stored in the memory along with the data word. Parity is checked when the word is taken out of the memory and placed in Q for transmission to other parts of the computer. If a parity error exists, the address at which the error occurred is stored in the E* register until such a time that the error can be analyzed. The address system also has provisions for checking parity. The address that is sent to the V register is checked against the parity of the P or U* registers each time a Variable Memory reference is made. If there is a parity error, the address in P* is stored in the E* register until such a time that the error can be analyzed.

Permanent Memory.—The Permanent Memory is a word organized, large capacity, non-destructive readout storage unit. Information is stored in the memory in a pattern of permanent magnets arranged on a card. A twistor matrix is used to sense the presence or absence of a magnet for the bits of a word being addressed. The unit contains 10,240 address locations each capable of storing a 24-bit instruction word or constant plus parity. Characteristics of the unit are high-speed operation, random access to stored data, and retention of data when power is turned off (non-volatility). The time required for one memory reference (the basic memory cycle) is five microseconds. Permanent Memory is used to store critical constants and instruction words. Because the program is stored in permanent magnets, the cards in which the magnets are embedded are called "program cards." These cards are inserted along columns of twistor wires so that the magnets are close enough to set up an external magnetic field in the twistor wires. The presence or absence of a magnet denotes a "0" or a "1." Thus the program can be changed by inserting new program cards containing a revised pattern of magnets. A block diagram of Permanent Memory and associated control circuitry is shown in FIGURE 3.

A read operation followed by a bias operation make up what is called the basic cycle in Permanent Memory. The duration and timing of these operations is controlled by the Permanent Memory control flip-flops and by clock pulses from the Main Pulse Distributor. Major sequence commands for the transmission of address information to the W register and for data transmission to and from the Z register are issued by the Control Section. W, the 14-bit address register, selects one of the 10,240 address locations during a Permanent Memory reference. The translators associated with W simultaneously enable X and Y group access switches (selectors) and line access switches (selectors) at opposite ends of the drive lines to permit the flow of current through the word access cores. Switching the word access cores enables data to be read out of Permanent Memory. The information stored on the program cards is not destroyed by the read operation (nondestructive readout), therefore a write operation is not necessary. However, since the word access cores change stages during a read operation, a bias current which is on continuously is used to switch the cores back to their original state. Once the address has been selected and the read operation initiated, data is transmitted from Permanent Memory to the rest of the computer through Z, the 24-bit transfer register.

In addition to transferring a 24-bit data word, the Z register also contains an additional bit for parity. Parity is stored in the memory along with the data word and is checked when the word is taken out of the memory and placed in Z for transmission to other parts of the computer. If a parity error is found, the address at which the error occurred is stored in the E* register until such a time that the error can be analyzed. The address system in Permanent Memory also has provisions for checking parity. The address that is sent to the W register is checked against the parity of the P or U* register each time a Permanent Memory reference is made. If there is a parity error, the address in P* is stored in the E* register until a time that the error can be analyzed.

Twistor Input Switch.—The Twistor Input Switch (Twistor-Input Unit or TIS) is a word-organized device that receives data from certain external units and makes it immediately available to the computer. The characteristics and principles of operation of the switching unit are similar to those of permanent Memory. However, since data from the external unit is continually changing, the permanent magnets that are used as storage devices in Permanent Memory are replaced by electromagnets in the Twistor Input Switch, so that information being transmitted from the unit can be changed at random. The Twistor Input Switch has 48 address locations each of which is capable of storing a 24-bit word plus parity. Characteristics of the unit are high-speed operation, random access to stored data, and retention of data when power is turned off (assuming the inputs remain energized). The time required for reference (basic cycle) is 5 microseconds.

As in Permanent Memory, a read operation followed by a bias operation make up what is called the basic cycle in the Twistor Input Switch. The duration and timing of these operations is controlled by the Permanent Memory control flip-flops and by clock pulses from the Main Pulse Distributor. Timing for the data input is controlled by the external unit and interrupts may be generated at any time. Access circuits similar to those used in Permanent Memory are used to provide the read current and bias current in the Twistor Input Switch. Access to the 48 address locations is supplied by the 14-bit W register. Once the address has been selected and the read operation initiated, data from TIS is gated into the Z register for transmission to the rest of the computer. The parity check in the Twistor Input Switch is identical to that in Permanent Memory.

1.2. *Arithmetic*

The Arithmetic Section performs the basic operations of addition, subtraction, multiplication, division, square root, shifting and masking. The principal circuits are the arithmetic registers, X, Q, and (A), the Arithmetic Sequence Control (ASC), and the Shift Register (K). The Q, A, and K Registers are each double rank registers, with the second rank denoted by *. The X register is single rank. In certain operations, A, and Q, and A* and Q* are combined to form double length registers. The extended registers are designated by the combinations AQ and A*Q* respectively. The numerical quantities of an arithmetic operation are manipulated in the arithmetic registers; while ASC and the shift register are concerned with control operations. All arithmetic operations are performed in fractional, two's complement, binary arithmetic. For details of this section, reference may be had to co-pending U.S. applications Serial No. 137,805, filed September 13, 1961, and Serial No. 134,945 filed August 30, 1961, now Patent No. 3,192,366 dated June 29, 1965.

X Register.—The X Register is a 24 bit, single rank register. It functions, both as a buffer register and as an arithmetic register. As a buffer register, it serves as the data transfer register linking the arithmetic section to other computer sections. As an arithmetic register, it holds the addend, subtrahend, multiplicand, divisor, and root during the associated arithmetic operations. The X Register has neither shifting nor complementing properties. The complement of X, however, as well as the value 2X may be gated to the adder.

A Register.—The A Register, a 24 bit accumulator, is the principal register of the arithmetic section. It consists of the first rank register, A, and the second rank register A* and associated adder. The A Register is involved in all arithmetic operations. It holds the augend and accumulated sum in addition, the minuend and difference in subtraction, the most significant bits of a product in multiplication, the most significant bits of a dividend and the remainder in division, and the radicand in square root. Shifting properties are provided during transfers from A* to A. During multiply and divide operations, A and A* are tied to Q and Q*, respectively, to form double length registers.

Q Register.—The Q Register functions as an auxiliary register during operations of multiplication, division, and square root. In multiplication it holds the multiplier and the 24 least significant bits of the product; in division, the least significant bits of the dividend and the quotient; in square root, the remainder. Shifting properties are provided during transmissions from Q* to Q. During multiply and divide operations, Q and Q* are tied to A and A*, respectively, to form double length registers.

Arithmetic Sequence Control. The ASC receives operation code enables from the Control Section and directs the sequence of operations involved in the arithmetic processes. During extended sequence operations (multiply, divide, square root, and shift) ASC operates with K to produce the iterations necessary to complete a designated instruction. An extended sequence is any instruction that requires more than one instruction time (5 microseconds) to execute. When an extended sequence is performed, ASC assumes control of the Arithmetic Section. Therefore, the Control Section of the computer is free to execute any subsequent instructions that do not require the use of the Arithmetic Section. The execution of those instructions that occur during an extended sequence and are arithmetic in nature (require the use of the Arithmetic Section) is delayed until the extended sequence is completed. Such a delay is called an arithmetic lockout.

K Register.—The K Register holds the shift count during the shift instruction and the iteration count during multiply, divide, and square root. As each shift or arithmetic iteration is performed, (K*) is transmitted to the counter and reduced in count. If a single place shift or iteration is performed, the count is reduced by one; if a two place shift is executed the count is reduced by two. When $K=0$ the sequence is terminated.

A, Q*, and Skip Enable Tests.*—This unit (1) checks the contents of A* to determine whether it equals 0 or −1, (2) checks the contents of Q* to determine whether it equals 0, and (3) checks the sign of A, A*, and X to determine whether A is greater than or equal to X, or whether A is less than X.

1.3. *Input/Output*

The Input/Output Section functions as a buffer for the exchange of data between the computer and the external equipment. Communication is provided on two channels, designated channel 1 and 2. Each operates independent of the other under direction of the control section. Channel 1 transfers data from the computer variable memory to critical external units. Channel 2 transfers data from the variable memory to non-critical external units and also exchanges data with the tape units and the typewriter. The principal circuits include the communication registers C1 and C2, the function registers F1 and F2, C1 and C2 Sequence controls, the D and L registers, and the Signal Data Recorder-Reproducer Control.

C1 and C2 Registers.—These registers, each 25 bits, form the data link between the computer and the external units. Although C1 is employed only as an output transfer register in the present system, the necessary hardware for an input operation may be included if desired.

F1 and F2 Registers.—The function registers F1 and F2, each 13 bits, are associated with C1 and C2, respectively. These registers receive the function code from the control section, translate it, and provide the control enables which designate the operation to be performed and unit which is to operate.

C1 and C2 Sequence Controls.—These circuits provide the proper timing for sequencing transfers of data.

D Register.—This 7 bit transfer register (including parity) functions as a buffer register during communication with the certain peripheral equipment. During output operation D receives 6 bit characters from C2 and transfers them to the typewriter or to tape units. During input operations D receives 7 bit characters from the tape units or typewriter and transfers them to C2.

L Register.—This 7 bit register (including parity) receives characters from the D register and converts "return to zero" data to "change on 1" data for recording on tape. It also is utilized in writing an end of record check mark, and in performing a longitudinal check on drop outs during a tape read operation.

Signal Data Recorder Reproducer Control.—These circuits control the operations of D and L during conversion of a 24 bit computer word to a 6 bit tape or typewriter character, and vice versa.

1.4. *Error*

The basic function of the Error Section is the detection of errors that occur in all sections of the computer. There are two basic types of errors, program-generated and hardware-generated. Program-generated errors are classed as intermittent errors, since they result from erroneous programming and once generated, will probably not appear again. Hardware-generated errors can be intermittent also; the result of marginal conditions in the hardware. When the hardware fails completely, the error becomes "solid." Solid errors are those errors that are duplicated each time the hardware that produces the error is used. When errors are detected, those physical portions of the computer (C-planes and/or D units) that caused the error are located, by translation of the error itself and additional programming.

When an error occurs, the error and an "indicating program address" are stored in memory, thus enabling future references to both the error and indicating address. The indicating program address establishes a relationship between an error and its occurrence in a program. When an error occurs, certain bits of the E register are set, thereby producing in that register a quantity that specifies an error. This quantity (error) is implied in the statement: the error is stored. The Principal circuits of the error section are the various Error check circuits, the Error Register (E) and the Error Address Register (E*). For details of this section, reference may be had to the above identified U.S. Application Serial No. 137,805.

E Register.—The E Register, 25 bits, receives error data from the detection hardware, translates the data, and produces error light indicator enables designating the D Unit and C plane at fault. It also issues error interrupts to the control section which then stores the error data in Variable Memory.

E Register.*—This Register, 14 bits, records the address of the instruction in process at the time the error occurred, or in the case of an error on a memory reference, it records the address of the word being referenced. This data is also stored in variable memory for subsequent analysis.

Error Check Circuits.—These circuits are located in all computer sections and check data transmissions, command signal generations, timing operation, etc. to set a particular bit or bits of the E Register. These are as follows. O and Z Parity Check circuits determine which, if any, bits are lost when, during memory references, words are transmitted between the Variable Memory and the O register or between the Permanent Memory and the Z register. R, OR Gates and U***10–23** Parity Check Circuits check parity of the R register during R memory read operations, and sense parity of the OR Gates during R Memory write operations. Also, these circuits sense the parity of U***10–23** so that, whenever required, reading of parity from this portion of the U* register is possible. Address System Parity Check Circuits detect the loss of bits and the generation of errors in the address system register; namely, U***10–23**, V, W, P, and P*. Extended Sequence Error Check Circuits determine whether the program time allocated to the extended-sequence instructions (Multiply, Divide, Square Root, and Shift) is exceeded. Critical Command Error Check Circuits determine whether commands that are critical to the computer arithmetic operations have been produced when required.

Error Indication and Control.—Error Control supervises the transmission of data between the E and E* registers. Also, Error Control supervises the execution of those portions of the Error Interupt program that are related to the storage of detected errors. The E register communicates with the Error Detection Circuits and the Variable Memory via the O register. The E register receives all errors that the computer is capable of sensing and then, by employing error-analyzing programs, translates the word in E to a visual indication of faulty D-units or C-planes. These programs light error indicators on those C-planes and/or D-units that are functioning improperly.

In summary, the E* register receives data from P*, V, and W whenever an Error Interrupt is generated. Since P*, V and W are address registers, E* will receive an address when an error occurs. This address specifies the general location in the program at which the error occurred. Also, E* communicates with Variable Memory via the O register, thus enabling the transmission of the contents of E to O during the Error Interrupt program. The Error Indication and Control portion of the Error Section receives indications of computer errors and other information (addresses) that is useful in the program diagnosis of these error indications. In addition, this function group includes the circuitry that controls the operation of the Error Section during the Error Interrupt program.

GENERAL

2.0 *Memory addresses*

FIGURES 2 and 3 are block diagrams of the variable memory and permanent memory access circuits, respectively. However, a detailed description of each of these figures is postponed until after the V register and translator of FIGURE 21 or the W register and translator of FIGURE 24 have been discussed. For the present, it is sufficient to note that a variable memory 14 binary bit address is in the range of values from

00 100 000 000 000

through 00 111 111 111 111, inclusive. By dividing this fourteen bit address into four groups of three binary bits each and one group of two binary bits, as shown, this variable memory address range may be more conveniently expressed in octal notation as 04000 through 07777, inclusive. A 14 binary bit permanent memory address is in the range of values from 01 000 000 000 000 through 11 011 111 111, 111, inclusive. The permanent memory address range expressed in octal fashion therefore 10000 through 33777, inclusive. The twister input switch portion of the memory section may be referenced by a 14 binary bit address in the range of from

11 111 111 010 000

through 11 111 111 111 111, inclusive. Again, this range may be expressed in octal fashion from 37720 through 37777, inclusive. If a data operand word is located in the accumulator register of the arithmetic unit (an "A" address) instead of in either the variable, permanent, or TIS memories, then this location is specified by a 14 binary bit address of 00 011 000 000 000, expressed in octal fashion as 03000. Also, octal numbers of 03XXX (where X designates any octal digit) can be used to indicate the "A" address. All other 14 bit binary combinations other than those specified above refer to nonexistent memory locations.

2.1. *Basic Circuit*

FIGURE 4*a* shows a schematic of a transistor circuit which may be utilized as the basic logic element of the system, while FIGURES 4*b*, 4*c*, and 4*d* indicate the symbols used in FIGURES 5 through 44 to signify the particular logical function of the basic element in a particular environment. For the logical combination of input signals, two or more PNP transistors **T1** and **T2** have their emitters grounded and their collectors connected to the center of a voltage divider consisting of resistors **R1** and **R2**, with a negative potential being applied to the other terminal of **R1** to reverse bias the transistor collector circuits in the normal fashion. The output terminal **4–10** from the circuit is derived from the common junction of the transistor collectors. Individual input signals are applied to input terminals **4–11** and **4–12** which are respectively connected via resistors **R3** and **R4** to the base electrodes of **T1** and **T2**. Capacitors **C1** and **C2** are also shunted with **R3** and **R4** respectively, to perform a speed up function. The input signal applied to either terminal **4–11** or **4–12** is binary in nature in that it has two discrete potential levels such as is shown in the figure. When the input signal is at the lower of its two potentials, the circuit is designed such that the base electrode of the associated transistor is more negative than its grounded emitter so that conventional current flows in its collector electrode in the direction of the arrow. This current flows through **R1** back to the negative potential so that the terminal **4–10** produces a positive going signal therefrom. Conversely, if a positive going input signal is applied to a terminal **4–11** or **4–12**, then the base electrode of the associated transistor is raised to reduce or terminate the current in its output collector circuit. This therefore drives terminal **4–10** toward the negative potential connected to the other terminal of **R1**, thus producing a negative going signal output. The ON condition of any one of the transistors associated with resistor **R1** causes a positive output from terminal **4–10** such as indicated in FIGURE 4*a*. A negative going output from **4–10** is only generated if positive going signals are applied to the base electrodes of all of the transistors in the circuit. Thus, the circuit of FIGURE 4*a* acts logically as an AND-NOT circuit for positive input signals, and as a NOR circuit for negative signals. Although two transistors **T1** and **T2** have been shown in FIGURE 4*a*, one transistor, or three transistors may be provided depending upon the number of input signals to be sensed by the circuit.

FIGURE 4*b* indicates the symbol utilized for the circuit of FIGURE 4*a* when it is to be used for detecting the coincidence of positive input signals. The character $\overline{\text{A}}$ placed in the middle of block **4–13** indicates the AND NOT function of the circuit, which generates a significant negative output only when positive going signals are applied to all of its input terminals. In FIGURE 4*c*, the character $\overline{\text{O}}$ placed in the middle of block **4–14** indicates that the basic circuit acts as a NOR circuit so that a significant positive output is obtained therefrom when any one of its input signals is negative. In FIGURE 4*d*, the symbol N in the center of block **4–15** indicates a NOT function of the basic circuit such that the output is the inverted input. The basic circuit of FIGURE 4*a* will provide the NOT function if it includes only one transistor.

FIGURE 4*e* shows the block symbol used for a bistable flip-flop while FIGURE 4*f* shows the composition of the bistable flip-flop using two of the basic circuits in their logical NOR function. In FIGURE 4*e*, a flip-flop **4–16** is represented as having one or more set input terminals and one or more clear input terminals, with one set output (1) and one clear output (0) therefrom. A negative going signal applied to any one of the set input terminals of a flip-flop causes that flip-flop to assume a state such that a positive signal is produced from its (1) output terminal. Conversely, a negative going signal applied to any one of the clear input terminals causes the flip-flop to assume a state such that a positive going signal is provided from its (0) output terminal. A flip-flop having these characteristics may be constructed as shown in FIGURE 4*f* from two of the basic circuits acting as NOR gates. Each of the $\overline{\text{O}}$ gates is respectively cross coupled by having its output connected to an input of the other as shown. Other inputs are provided to each of the $\overline{\text{O}}$ gates **4–18** and **4–17** depending upon the number of set and clear input terminals required, respectively.

The operation of the logic configuration of FIGURE 4*f* is as follows, assuming that $\overline{\text{O}}$**4–17** receives negative "clear" signals as required, while $\overline{\text{O}}$**4–18** receives negative "set" signals as required. When neither a negative "clear" nor a negative "set" signal is to be applied to its associated input, the potentials of these inputs are all held positive. Assume first that a negative going "set" signal is temporarily applied to one of the set terminals of $\overline{\text{O}}$**4–18**. As illustrated in FIGURE 4*a*, any negative going signal to the basic logic circuit produces a positive output therefrom. The positive output produced by $\overline{\text{O}}$**4–18** is applied to one of the input terminals of $\overline{\text{O}}$**4–17**, with the signals applied to the other clear input terminals of this $\overline{\text{O}}$ gate also being positive in the absence of a "clear" input signal. Therefore, in this situation, positive signals are applied to all input terminals of $\overline{\text{O}}$**4–17** such that it produces a negative output signal which in turn is applied to one of the inputs of $\overline{\text{O}}$**4–18**. This negative signal from $\overline{\text{O}}$**4–17** maintains a positive output from $\overline{\text{O}}$**4–18** even after the temporary negative "set" signal disappears. Thus, the flip-flop is set to a stable state which produces a positive output from $\overline{\text{O}}$**4–18** and a negative output from $\overline{\text{O}}$**4–17**. In order to change stable states, a negative "clear" input must be temporarily applied to one of the clear input terminals of $\overline{\text{O}}$**4–17**, which thus produces a positive output therefrom. This positive output is applied to $\overline{\text{O}}$**4–18** further having positive signals applied to all of its set input terminals in the absence of a significant negative "set" signal. Therefore, the output from $\overline{\text{O}}$**4–18** becomes negative which in turn maintains the positive signals from $\overline{\text{O}}$**4–17** until a negative "set" signal is once again applied to $\overline{\text{O}}$**4–18**.

DETAILED FIGURE DESCRIPTION

In the remaining FIGURES 5 through 44, all $\overline{\text{A}}$ gates, $\overline{\text{O}}$ gates, N gates, ond control flip-flops are numbered in a manner to indicate the figure in which they are shown. Thus, the digit or digits to the left of the hyphen represents the figure number. For example, the term "$\overline{\text{A}}$**27–18**" refers to an $\overline{\text{A}}$ gate in FIGURE 27. Likewise, the term "$\overline{\text{O}}$**27–16**" refers to an $\overline{\text{O}}$ gate in FIGURE 27. The term "**FF27–10**" refers to the X shaper flip-flop in FIGURE 27. However, those flip-flops which comprise stages in any one of the various registers are designated only by the register letter followed by a digit or digits specifying its position in the register; i.e., **P01** (or **P1**); **U*10**, etc.

All signals entering a figure are specified by a word or words (sometimes abbreviated) usually indicating its function, as well as a digit or digits enclosed in parentheses for indicating the figure from which that signal is generated. The exceptions to this are if the signals are derived from either the set or clear output terminals of a flip-flop in a register, or if the signal is generated from a unit shown in FIGURE 1 which is not disclosed in detail in the instant application. In the case of the former, the register stage is identified but the figure in which it appears is not given.

3.0. *Program Address Counter*

FIGURE 5 shows the **P** register which is a 14 bit register used to hold the address of the next instruction to be performed. Fourteen flip-flop stages **P00** through **P13** are provided in the register although for the sake of clarity, only eight of these stages are shown, with stages **P05** through **P10** being omitted. A 14 bit address may be entered into the P register from three different locations within the control unit. During normal instruction address sequencing, the address is inserted into P register from the address counter of FIGURE 7 via the conductors marked SET **P00** (7) through SET **P13** (7). As will be subsequently explained, the address counter has incremented the immediately preceding instruction address by one so that this procedure results in the instructions being normally withdrawn in sequence from the memory. The P register may also be loaded directly from the P* register via the conductors labeled SET **P00** (6) through SET **P13** (6). This function is performed during the execution of a Repeat instruction 40, or if an extended sequence arithmetic operation is currently being performed by the machine and certain other conditions are met. The third location from which the P register may be loaded is the U* register stages **10–23** which contain the address portion of the current instruction. This operation is generally performed during the execution of any one of the jump instructions 34, 35, or 36, and also during the execution of the Repeat instruction 40. Information from U* **10–23** is gated to **P00–13** via $\overline{\text{A}}$ gates **5–10** through **5–23** which are respectively conditioned by an output from **$\overline{\text{O}}$5–24**. **$\overline{\text{O}}$5–24** in turn receives signals from **$\overline{\text{A}}$5–25** and **$\overline{\text{A}}$5–26**, either of which is sufficient to make the transfer. **$\overline{\text{A}}$5–25** is conditioned at MP3 time and produces an output if a memory jump instruction is being currently performed. **$\overline{\text{A}}$5–26** is gated at MP9 time and generates an output during execution of the Repeat instruction 40.

P register stages 0 through 13 may be cleared to an empty condition by any one of a number of ways via **$\overline{\text{O}}$5–28** and **$\overline{\text{O}}$5–33** whose outputs are respectively inverted by **N5–29** and **N5–34**. For example, prior to entry of an address from the address counter, a Clear P signal is applied to **$\overline{\text{O}}$5–28** and **$\overline{\text{O}}$5–33** from the memory access control unit in FIGURE 39. A Clear P signal from the skip and arithmetic lock-out control unit of FIGURE 40 is also applied to the latter two $\overline{\text{O}}$ gates during a skip operation. Clear signals for the P register during execution of the Repeat instruction **40** are also provided by **$\overline{\text{A}}$5–30** and **$\overline{\text{A}}$5–32**. In like fashion, **$\overline{\text{A}}$5–31** is responsive to a timing signal MP2, together with indication of a jump instruction, in order to provide a clear signal to the P register.

All of the above identified clear signals for the P register are applied simultaneously to **$\overline{\text{O}}$5–28** and **$\overline{\text{O}}$5–33**. It will be noted that the output from **$\overline{\text{O}}$5–28** is applied to clear all of the P register stages except **P02**, which is cleared only by the output from **$\overline{\text{O}}$5–33**. During the initial start-up of the machine, a Set P=04000 signal is applied only to **$\overline{\text{O}}$5–28** so as to clear all but the **P02** stage. In turn, a reset P=04000 signal is applied to set the **P02** stage as indicated in FIGURE 5. This operation causes a memory address of 04000 to be inserted into the P register which is the memory location of the first address to be performed in the program. Normally, however, all other clear signals result in outputs being emitted from both **$\overline{\text{O}}$5–28** and **$\overline{\text{O}}$5–33** so that all register stages of P are cleared at the same time.

FIGURE 5 also shows the address range translator in which the address of the next instruction held in P register is translated to determine whether it is a variable or a permanent memory address, or whether an illegal P address has been computed by the program address counter. The address range translator need only sample the binary bits held by the P register stages **P00**, **P01**, and **P02**. **P00** and **P01** respectively represent the binary coefficients 2^1 and 2^0 which make up the highest order octal digit which may have a value of from 0 to 3. **P02** represents the binary coefficient 2^3 making up the second highest order octal digit which may have a value of from 0 to 7. Thus, if both stages **P00** and **P01** are in their cleared condition, the highest order octal address digit is 0, while if both of these stages are set, the highest order octal digit has a value of 3. In like manner, if **P02** is in its cleared condition, the second highest order octal digit may have a value of from 0 to 3 while if the stage is in its set condition, the second highest order octal digit has a value of 4 or above.

The manner in which the address range translator operates may be clearly appreciated from Table 1 given below.

TABLE 1

P Register	Mem. Adr.	V. Mem.	W. Mem.	Illegal
$\overline{P0}\cdot\overline{P1}\cdot\overline{P2}$	00XXX–03XXX	No	No	Yes
$\overline{P0}\cdot\overline{P1}\cdot P2$	04XXX–07XXX	Yes	No	No*
$\overline{P0}\cdot P1\cdot\overline{P2}$	10XXX–13XXX	No	Yes	No
$\overline{P0}\cdot P1\cdot P2$	14XXX–17XXX	No	Yes	No
$P0\cdot\overline{P1}\cdot\overline{P2}$	20XXX–23XXX	No	Yes	No
$P0\cdot\overline{P1}\cdot P2$	24XXX–27XXX	No	Yes	No
$P0\cdot P1\cdot\overline{P2}$	30XXX–33XXX	No	Yes	No
$P0\cdot P1\cdot P2$	34XXX–37XXX	No	No	Yes

*Yes, if in real time mode.

If all three of the sensed stages **P0**, **P1**, and **P2** are cleared, each of the signals **$\overline{\text{P0}}$**, **$\overline{\text{P1}}$**, and **$\overline{\text{P2}}$** is positive and switch at **$\overline{\text{A}}$5–42** to provide a negative output therefrom. With this configuration of P register stages, the address contained in the P register is in the range of from 00000 to 03777, with the X values in Table 1 indicating that the digit in that octal order may have any value from 0 to 7. In accordance with the previous address range given before, it is seen that none of the addresses represented will reference either the Variable or the Permanent (V or W) memory so that the address held in the P register at this time is illegal. The negative output from **$\overline{\text{A}}$5–42** is applied to **$\overline{\text{O}}$5–47** which results in a positive output therefrom, which in turn switches with MP1 and a signal indicating the absence of a memory jump instruction to provide a negative output from **$\overline{\text{A}}$5–50**.

In the event that **P0** and **P1** are both cleared while **P2** is set, the signals **$\overline{\text{P0}}$**, **$\overline{\text{P1}}$**, and **P2** are positive. The address now contained in the P register will thereby have a value of from 04XXX through 07XXX which is a variable memory address. Since **$\overline{\text{P2}}$** is now negative, there is no significant output from **$\overline{\text{A}}$5–42** which means that its output remains positive. **$\overline{\text{P0}}$** and **$\overline{\text{P1}}$** switch at **$\overline{\text{A}}$5–43** together with the output from **$\overline{\text{A}}$5–42** to provide a negative output therefrom, thus indicating that the condition $\overline{P0}\cdot\overline{P1}\cdot P2$ has been met. The negative output from **$\overline{\text{A}}$5–43** is inverted via **N5–44** and applied to **$\overline{\text{A}}$5–41** as a positive signal. However, in order to get a significant output from **$\overline{\text{A}}$5–41**, positive signals P To V Or W Enable, together with a signal indicating that the system is not in real time mode, must also be applied thereto. If such is the case, then a negative output is derived from **$\overline{\text{A}}$5–41** which, when inverted via **N5–46**, becomes a positive signal P=V/Adr. This signal is used to gate the contents of the P register to the variable memory address register V in a manner subsequently to be described. If the system is not operating in the real time mode, then a variable memory address in P register is a valid one. However, if a system is operating in the real time mode, then a variable memory address in P becomes illegal. This condition is indicated by switching the output of **N5–44** against a positive Real Time Mode signal at **$\overline{\text{A}}$5–48** to provide an input to **$\overline{\text{O}}$5–47**. This in turn provides a positive input to **$\overline{\text{A}}$5–50** which may switch with the other positive signals to generate an error signal. In such a case, the signal Not Real

Time Mode applied to one input of $\overline{A}$**5–41** is negative in order to prevent the positive signal P=V/Adr. from being generated at this time.

Jumping now to the last line in Table **1**, if all three stages **P0**, **P1**, and **P2** are set, then the address contained in the P register is in the range of from 34XXX through 37XXX none of which is a valid memory location for instruction referencing. It should here be noted that the Twistor Input Memory with addresses 37720 through 37777, acts as a buffer between certain external units and the computer for data only, but not instructions. This combination of the P register stages is sensed by $\overline{A}$**5–49** which supplies a signal input to $\overline{O}$**5–47** to thereby indicate an illegal address contained in the P register.

Returning now to the remaining combinations available for the three P register stages being considered, it is noted that the combinations indicated in the third through the seventh line of Table 1 represent that the P register contains a memory address in the range of from 10XXX to 33XXX, which are valid permanent memory addresses for instruction referencing. The translator indicates the valid permanent memory address by a method of exclusion, which means that if no illegal or variable memory addresses are noted, then the address must be a valid permanent memory address. This signal is generated by $\overline{O}$**5–40** which receives inputs from two illegal address sense gates $\overline{A}$**5–42** and $\overline{A}$**5–49**, as well as the variable memory address sense gate $\overline{A}$**5–43**. Thus, if none of these gates has supplied a negative output therefrom, the output from $\overline{O}$**5–40** remains negative which, when inverted by **N5–45** becomes the positive signal P=W Adr. $\overline{O}$**5–40** also must receive the positive signal P To V Or W Enable from FIGURE 41 in order to insure that a negative output continues therefrom at this time. If, however, the P register address is illegal or if it represents a variable memory address, then a positive signal will be produced from $\overline{O}$**5–40** which in turn causes the negative signal from **N5–45**.

As before noted, any signal from $\overline{O}$**5–47** switches at $\overline{A}$**5–50** with a positive MP1 and a signal indicating the presence or absence of a current jump instruction in the instruction register. The indication of an Illegal P Address Error is therefore made on MP1 if no jump instruction is being performed. The appearance of this signal from $\overline{A}$**5–50** prevents any further computation of addresses by the program address counter, which holds the illegal address until the P register is cleared manually (by a signal not shown) from the computer control panel. However, if a jump instruction is actually being performed at this time, then the signal from FIGURE 39 is negative which prevents its negative output from being generated by $\overline{A}$**5–50**. No indication of error is required during execution of a jump instruction inasmuch as when the jump is taken, the illegal address in P is lost and is never used for a memory reference. Furthermore, in connection with the address range translator, it should be understood that the positive signal **P** to V Or W Enable, which is applied to $\overline{A}$**5–41** and $\overline{O}$**5–40**, is never positive if the next instruction is to be locked out. This operation will be clarified in subsequent discussion.

The P* register shown in FIGURE 6 is used primarily as storage for a memory instruction address which is to be incremented by the address counter of the PAC unit. Furthermore, it is also employed during the execution of a repeated instruction to store that address of the instruction next to be performed after the execution of the repeated instruction has been terminated. After an address in the **P** register has been used to reference the memory, the **P** register address is transferred to **P*** register, after which it is transferred from the **P*** register back into the **P** register via the address counter which selectively adds either "1" or "2" thereto. In FIGURE 6, the P* register is seen to comprise fourteen flip-flop stages **P*00** through **P*13** into which may be gated an address from the P register via $\overline{A}$ gates **6–10** through **6–27**. The gating signal applied to the other inputs of the $\overline{A}$ gates is derived from $\overline{O}$**6–24**. During the normal sequence of operations, a signal P To P* is derived from FIGURE 39 which transfers the P register address into P* register at MP4 time. A transfer of P register into P* must also occur during the execution of the Repeat instruction **40** and this is accomplished by the switching of the Command Repeat Instruction and MP7 and $\overline{A}$**6–25** which in turn provides a significant output to $\overline{O}$**6–24**. Thus, if a negative signal is applied to either of the two inputs to $\overline{O}$**6–24**, a positive signal is generated therefrom which gates positive signals from the set stages of the P register so as to set correspondingly numbered stages of the P* register via the indicated $\overline{A}$ gates. The P* register is initially cleared by one of two negative signals appearing as inputs to $\overline{O}$**6–26** whose output is inverted by **N6–27** and applied to the clear input of the P* register stages. This is performed by either a Clear P* signal from FIGURE 39, or an MP6 signal switching with Repeat Instruction at $\overline{A}$**6–28**. These two clear times occur respectively before the gating times indicated by the $\overline{O}$**6–24**.

An output from both sides of each P* register stage is taken directly to the address counter next to be described. Furthermore, as previously noted, there are times in which it is desired to gate the address in P* directly to P register without incrementing same by 1 or 2. The cleared condition of each P* register stage, thus indicating a binary "0" bit, is gated via a set of $\overline{A}$ gates **6–30** through **6–43** to the clear inputs of the correspondingly numbered P register stages in FIGURE 5. In like fashion, the set indications of the P* register stages, indicating binary 1, are gated via the $\overline{A}$ gates **6–50** through **6–63** to the set terminals of the corresponding numbered P register stages. The gating time is determined by the positive signal P* To P from FIGURE 41 which is generated during the execution of a Repeat Instruction 40.

The address counter of the PAC unit is shown in FIGURE 7. This circuit adds either one or two to the address held in P* and places the result into the P register for use in withdrawing the next instruction from memory. Since the least significant bit of the memory address held in the P* register is stored by **P*13**, 1 or 2 is added to the value here with appropriate carries being propagated from **P*13** to **P*01**. Just prior to the incrementing of the P* register contents and its transfer to P register, the P register stages are all cleared as previously described so that each stage is considered to contain a binary bit of 0 value. Thereafter, if a positive signal is enabled from any $\overline{O}$ gates **7–10** through **7–23**, such positive signals switch at respective ones of $\overline{A}$ gates **7–24** through **7–37** with a timing signal in order to set the associated P stages **P00** through **P13** to values indicative of the newly generated address. The timing signal for conditioning $\overline{A}$ gates **7–24** through **7–37** is generated from $\overline{O}$**7–38** due to the presence of any one of the three signals P*+N To P from FIGURE 40, P*+N To P from FIGURE 39, or a negative output from $\overline{A}$**7–104**. The last timing signal occurs at MP4 time when a Repeat instruction 40 is being executed, while the two former signals respectively occur during MP6 time for the normal overlap operation and during MP15 time during execution of a skip instruction.

The setting of a P register stage by the address counter, i.e. the placing of a 1 bit into a register stage P_n, is governed by the following equation:

$$P_n = P_n{}^*\overline{C_{n+1}} + \overline{P_n}{}^*C_{n+1}$$

In the above, P_n represents any one of the P stages **0** through **13**, $P_n{}^*$ represents the correspondingly numbered P* stage, C_{n+1} represents a carry of 1 from the adjacent higher numbered register stage (of lower binary order significance), and C_{n+1} represents a carry of 0 from said adjacent higher numbered register stage. This equation thus signifies that if P^*_n stage contains a binary 0, the addition to this value of a carry of 1 from the adjacent stage results in a binary 1 being placed into the correspondingly numbered P_n stage. Likewise, if the $P_n{}^*$ stage contains a binary 1, an adjacent lower order carry of 0 does not change this value, which is consequently placed into the P_n stage. The combination of a 0 bit in $P_n{}^*$ and a 0 carry, or the combination of a $P^*_n=1$ bit and a 1 carry results in a binary 0 being placed into the corresponding P_n stage. However, inasmuch as the P register is initially cleared at commencement of the addition process, there is no need to provide a special Clear P_n signal from the address counter.

The above equation is mechanized as follows in the address counter of FIGURE 7. Ā gates **7–39** through **7–52** each has an input thereto from the set side of a respective one of the P* register stages **1** through **13**. Ā gates **7–53** through **7–66** likewise have an input thereto from the clear terminal of respective one the P* stages **1** through **13**. The carry value from the adjacent lower binary order stage of the address counter (the adjacent higher numbered stage) is provided by Ā gates **7–71** through **7–82** for **P0–11**, and by **N7–97** for position **12**. The initial carry to the lowest binary order position **P*13** is provided by the PAC ADD Flip-flop **7–112** which, when cleared, increments the P* address by 1, and when set increments P* address by 2. In the absence of a significant negative output from Ā gates **7–71** through **7–82**, it is seen that only Ā gates **7–39** through **7–50** will be enabled so as to gate through a set signal, if any, from its associated P* stage. Thus, the absence of a negative output from an Ā gate **7–71** through **7–82**, and the consequent positive output therefrom, indicates that a carry of 0 is received from the adjacent lower order stage of the counter so that the first term of the above equation is satisfied. A negative output from an Ā gate **7–39** through **7–51** is applied to the associated Ō gate **7–10** through **7–21** in order to set the correspondingly associated P_n stage. Conversely, if any one of the Ā gates **7–71** through **7–82** has a negative output therefrom, thereby indicating a carry of 1 from the adjacent lower order stage, then the associated Ā gate **7–39** through **7–50** will be conditioned while the associated N gates **7–84** through **7–95** will enable associated Ā gate **7–53** through **7–64** to pass the positive signal, if any, from the clear output terminal of the associated P^*_n stage. Thus, the latter circuits provide the logical function indicated by the second term of the above equation in that a significant negative output from any of the Ā gates **7–53** through **7–64** causes a set pulse to be applied to the associated P_n stage.

In connection with address counter stages **12** and **13**, which respectively determine the setting of **P12** and **P13**, the operation is somewhat different. Assuming that PAC ADD **7–112** is in its clear condition, Ā**7–66** is enabled to provide a significant negative output only if **P*13** is cleared. The output from Ā**7–66**, when applied to Ō**7–23** sets **P13** to a binary 1 condition. This follows because the addition of 1 to a 0 bit contained in **P*13** results in a 1 bit for the newly developed address placed into the P register. $\overline{\mathbf{P^*13}}$ and $\overline{\text{PAC ADD}}$ likewise switch at Ā**7–83** which provides an input to **N7–97**. Under the assumption that $\overline{\mathbf{P^*13}}$ is positive, it is therefore seen that a significant negative output from Ā**7–83** is obtained which, when inverted by **N7–97**, enables Ā**7–51** to pass a positive $\overline{\text{P*12}}$ if such exists. Ā**7–83** and **N7–97** thereby determine the nature of the carry generated from the lowest order stage of the address counter due to the addition of the bit contained in **P*13** and the 1 introduced by **FF7–112**. If **P*13** contains a binary 0, then obviously no carry will be generated. Therefore, in accordance with the first term of the above equation, the absence of a carry to stage **12** results in **P12** being set if **P*12** contains a binary 1. Assuming, however, that **P*13** holds a binary 1, then **P13** must be filled with a binary 0 and a carry generated for entry into the 12th stage of the address counter. As noted in FIGURE 7, a negative signal $\overline{\text{P*13}}$ does not allow Ā**7–66** to be enabled, which thus prevents the setting of **P13**. Likewise, the negative signal $\overline{\text{P*13}}$ applied to Ā**7–83** does not generate a significant negative signal therefrom so that the output from Ā**7–83** remains high. A positive signal from this Ā gate thus represents that a carry has been generated in the lowest order stage **13** of the counter which must be applied to stage **12** of the counter. A positive input to **N7–97** becomes negative therefrom which in turn conditions only Ā**7–65**. Thus, a Set P12 signal is generated from Ā**7–36** only if **P*12** contains a binary 0, inasmuch as this binary 0 plus the carry of 1 must result in a 1 bit being placed in **P12**.

The counter of FIGURE 7 employs simultaneous carry circuitry inasmuch as it is not necessary for the carry to propagate successively through each stage of the address counter. Instead, the carry input to each stage is simultaneously determined with the setting of the P* register stages. This is accomplished by dividing the address counter stages into groups and by examining the contents of the P* stages associated with each group to determine if any has a binary 0 bit which might absorb a carry of 1 coming from the adjacent lower order stage. If such occurs, then a carry of 1 will not be generated from the highest order address counter stage in any group. If, however, all of the P* stages in a group contain a binary 1, then an input carry of 1 to the lowest order stage in that group will not be absorbed and thus appears as an output carry of 1 to the lowest order stage of the next higher group of address counter stages. The gating circuits for performing this comparison are Ā gates **7–71** through **7–82**, previously described, as well as Ā gates **7–100** and **7–102**. For example, a significant negative output from Ā**7–83** indicates that no carry is generated within the stage **13**. Therefore, there will be no change in any of the bits contained in **P*12** through **P*0** and these will be transferred directly to the corresponding stages of the P register. The negative output from Ā**7–83** thereby disables Ā**7–81** and **7–82** so that positive outputs, indicating a carry of **0** are obtained from each. The positive output from Ā**7–81** is inverted via **N7–94** and applied as a negative input to Ā gates **7–71** through **7–80**, thus indicating that a positive output will be derived from each. On the other hand, if Ā**7–83** is not enabled due to the fact that **P*13** contains a binary 1, then the positive output therefrom will not by itself disable the last mentioned set of AND gates. A positive output from Ā**7–83** is applied to Ā**7–82** to which is also connected the set output terminal of **P*12**. If the latter terminal is positive, thus indicating the presence of a binary 1 in **P*12**, then Ā**7–82** is unable to provide a negative output signal therefrom. This therefore indicates that the addition of a carry of 1 to the binary 1 in **P*12** in stage **12** of the address counter results in carry **1** output to stage **11** of the address counter which in turn must be added to the content of **P*11**.

Ā**7–81** compares the contents of **P*11** with the contens of **P*12** and Ā**7–83** in order to determine whether a carry of **1** is to be sent from stage **11** to stage **10** of the address counter. Thus, it may be seen that if both **P*11** and **P*12** contain binary 1's, the presence of a carry 1 from stage **13** to stage **12** of the address counter cannot be absorbed within either of the stages **11** or **12** and so must result in a carry 1 to stage **10**. This operation is indicated below, where the contents of **P*11 through** P*13 are indicated as being binary 1's, and the numerals above the arrows between the sum digits indicate the value of the carry generated between adjacent address counter stages.

```
P*10  P*11  P*12  P*13
  X     1     1     1
                    1  PAC FF
-----------------------
     1     1     1

 ←     0 ←   0 ←    0
```

If a negative signal appears from $\overline{\mathrm{A}}$**7–81**, thus indicating a carry input to address counter stage **10**, a positive signal appears from N**7–94** which is applied to $\overline{\mathrm{A}}$ gates **7–71** through **7–80**. $\overline{\mathrm{A}}$**7–80** likewise samples the content of P***10** in order to determine whether the addition of a carry 1 to P***10** will result in carry 1 to stage **0** of the address counter. Thus, if P***10** is positive, a significant negative output from $\overline{\mathrm{A}}$**7–80** appears. The same two inputs to $\overline{\mathrm{A}}$**7–80** are also applied to $\overline{\mathrm{A}}$**7–79** which further senses the condition of P***09**. Since the two positive inputs to $\overline{\mathrm{A}}$**7–80** indicate that a carry of 1 is to be added to the content of P***09**, $\overline{\mathrm{A}}$**7–79** may thereby recognize whether an output carry will be generated from address counter stage **9** to address counter stage **9**. In like fashion, $\overline{\mathrm{A}}$ stages **7–78** and **7–77** respectively sense lower order **P*** stages as well as the input carry to the group from N–**94** in order to determine whether carries will be applied to their associated address counter stages.

The set terminals of P* stages **6** through **10** are switched at $\overline{\mathrm{A}}$**7–100** whose ouput is inverted via N**7–101** and applied to $\overline{\mathrm{A}}$ gates **7–71** through **7–76**. $\overline{\mathrm{A}}$**7–76** also senses the set condition of P***06** as well as the carry generated from N**7–94**. Thus, if P* stages **6–10** have binary 1 bits, and if an input carry is applied to stage **10** of the address counter, then this input carry will not be absorbed by any of the register stages **P6** through **P10** so as to result in an input carry 1 also being applied to stage **5** of the address counter. This condition is detected by $\overline{\mathrm{A}}$**7–76**, however, if any one of the P* stages **6–10** contains a binary 0, then at least one input of $\overline{\mathrm{A}}$**7–76** will be negative so as to prevent enabling thereof, thus indicating that there is a carry of 0 to stage **5** of the address counter. Therefore, an input carry to stage **10** of the counter, although also applied to $\overline{\mathrm{A}}$**7–76**, will be unable to generate an output therefrom due to the fact that it will be absorbed within the address counter stage containing said binary 0.

The generation of carries for counter stages **0–4** is now believed to be obvious in view of the above detailed description. $\overline{\mathrm{A}}$**7–102** samples the values contained in P* stages **3**, **4**, **5**, and **6** and applies its inverted output to $\overline{\mathrm{A}}$**7–72**. The latter $\overline{\mathrm{A}}$ gate also senses the value of the bits contained in P* registers **7** through **10** because it receives the output from N**7–101**. Thus, $\overline{\mathrm{A}}$**7–72** is able to detect if all of the P* stages **2** through **10** have binary 1's contained therein, as well as the value of the input carry applied to stage **10** of the address counter from $\overline{\mathrm{A}}$**7–81**. Thus, if P* stages **2** through **10** each contains a binary 1, an input carry to stage **10** will not be absorbed therein and so would create a carry 1 from address counter stage **2** to address counter stage **1**. $\overline{\mathrm{A}}$7–71 has similar inputs in addition to an input from P*1. Therefore, it may be seen that the address counter in FIGURE 7 conditions the respective $\overline{\mathrm{O}}$ gates **7–10** through **7–23** to emit their proper output signals so that each may be simultaneously gated by a timing signal emerging from $\overline{\mathrm{O}}$**7–38** without there being any carry delay within the circuit.

As previously mentioned, PAC ADD FF**7–112** when cleared increments the address in the P* register by a value of 1. When this flip-flop is set, the P* address is incremented by a value of 2. Consequently, in view of the well known principles of binary addition, the sum bit placed into the **P13** stage has the same value as that held in P***13**, with a carry always being generated within stage **13** of the counter and passing to the higher order stages. Thus, when flip-flop **7–112** is set, $\overline{\mathrm{A}}$**7–52** is enabled to supply a set signal to the P register only if P***13** is set, otherwise, if P–**13** is cleared, then P13 remains cleared. Furthermore, $\overline{\mathrm{A}}$**7–83** must always generate a positive signal therefrom, indicating carry of **1**, inasmuch as it receives one input from the clear side of FF**7–12** which is negative during the addition of 2. Therefore, a carry will always be produced by the circuitry of FIGURE 7 between stages **13** and the high order stages of the counter.

As has previously been described, instructions are normally withdrawn from sequential memory addresses during the regular operation of the system. In such case, the address in P register is sent to the selected memory address register V or W at the beginning of a machine cycle, after which the P register is immediately incremented by 1. Flip-flop **7–112** is thus unconditionally cleared at MP1 and MP6 times via respective N gates **7–119** and **7–118**. However, during execution of any one of the skip instructions, it may be necessary to skip the next sequential instruction address if the skip condition is met. Thus, provision is made for setting FF**7–112** during execution of a skip instruction by means of $\overline{\mathrm{O}}$**7–115** and $\overline{\mathrm{A}}$**7–114**. Also, the execution of the Repeat Instruction 40 requires that flip-flop **7–112** be set. This function is provided by $\overline{\mathrm{A}}$**7–113** which also sets the Instruction Time Extension Repeat flip-flop **7–116** for use in a manner subsequently to be discussed.

3.1. *Instruction Registers and Command Translators.*—FIGURE 8 shows stages **0** through **5** of the U register into which is inserted the 6 bit operation code of an instruction word. As previously noted, a 24 bit instruction word may be obtained from either permanent memory, or from variable memory if the system is not operated in the guide mode. $\overline{\mathrm{A}}$ gates **8–10** through **8–15** consequently can be enabled by a signal Z To U0–23 from FIGURE 39 at an appropriate time to sample the contents of the Z transfer register associated with the permanent memory. In like fashion, $\overline{\mathrm{A}}$ gates A–**16** through **A–21** can be enabled by O To U**0–23** from FIGURE 39 in order to gate information into the U register from the O transfer register associated with the variable memory. Y**0** through U5 may also have forced therein two different operation codes by circuitry later to be described in detail. For example, in input-output data transfer operation is to be executed, a negative signal appearing from FIGURE 43 sets **U00**, **U03** and **U05** which in turn are translated to enable the execution of the input-output data transfer. In like fashion, when the main program must be interrupted for one of several reasons in order that a sub-routine may be initiated, a negative signal appearing from FIGURE 43 sets **U00**, **U03**, and **U04** such that U**0** through **5** stages contain an operation code enabling the execution of an interrupt sub-routine. Data transfer and interrupt operations will be described in detail at a later time. Prior to the setting of any of the stages **U0** through **5**, they are cleared by means of a clear **U0–05** signal from FIGURE 40 via **NA–22** so that those stages remaining clear after a subsequent set operation indicate the presence of a binary 0.

The 6 bit operation code of an instruction word may be divided into two 3 bit groups, each representing an octal digit. Since any three binary bit group has eight unique combinations, it is therefore seen that the maximum octal digit value is 7. Thus, an octal order may contain a digit in the range of from 0 to 7, while two octal order have a range from 00 to 77. U stages **0** through **2** hold the first octal digit (higher order) and stages U3 through **5** hold the second octal digit (lower order). Table 2 shows the instruction repertoire of the present system.

Table 2.—THE REPERTOIRE

Codes		Name	Function
(1)	(2)	(3)	(4)
NOP	00	No operation	Housekeeping Commands only.
ADD	01	Add	A+(T) to A.
CAD	02	Clear Add	(T) to A.
LA	03	Load A	(T) to A (no parity check).
SUB	04	Subtract	A−(T) to A.
CSB	05	Clear Subtract	−(T) to A.
MK	06	Mask	Q O (T) to A.
ES	10	Equality Skip	if A = (T), skip NI.
NES	11	Non-equality Skip	if A ≠ (T), skip NI.
LS	12	Less Skip	if A < (T), skip NI.
GS	13	Greater or Equal Skip	if A ≥ (T), skip NI.
MPY	14	Multiply	A . (T) to AQ.
DIV	15	Divide	AQ/(T) to A, remainder to Q.
SOR	16	Square Root	$\sqrt{(T)}$ to A, remainder to Q.
SE	20	Store E	E to T.
SF1	21	Store F1	F1 to T.
SF2	22	Store F2	F2 to T.
SA	23	Store A	A to T.
SQ	24	Store Q	Q to T.
SR	25	Store R	R_j to S (bits 10-23).
LR	26	Load R	(S) to R (bits 10-23).
IR	30	Increment R	T to R_j.
IF1	31	Initiate F1	T to F1 (initiate ch. 1I/O sequence).
IF2	32	Initiate F2	T to F2 (initiate ch. 2I/O sequence).
LC	33	Load Constant	S to R_j.
UCJ	34	Unconditional Jump	T to P.
SRJ	35	Subroutine Jump	P to R_j, S to P.
IJ	36	Index Jump	if $R_j \neq -1$, S to P; R_j +1 to R_g.
RP	40	Repeat	Initiate Repeat of NI R7 times.
SS	41	Selective Sense	if Sn = Mn, stop or skip per Mn.
SH	42	Shift	Shift A and/or Q by k places.
LQ	43	Load Q	(T) to Q.
LE	44	Load E	(T) to E.
IO	45	I/O Data Transfer	Cn to $\underline{O}$ to $\underline{O}$ to Cn, as per Fn.
INT	46	Interrupt	P to R_j, S to P.

In this table, columns 1 and 2 indicate the abbreviation for the instruction code and its two octal digit representation, respectively. Column 3 gives the name of the instruction code, while column 4 indicates the function performed in response to the associated instruction code. Referring to column 2, therefore, the left higher order octal digit is thereby represented by the 3 binary bits contained in **U0** through **2** while the right lower order octal digit is represented by the three binary bits contained in **U03** through **05**.

Prior to the generation of a one or more commands necessary for the execution of an instruction, an intermediate operation is performed in FIGURE 8 which converts the 6 outputs from **U0** through **5** into a single output signal unique to the operation code. In addition to the single output signal, several other signals may be generated which are common to others of the operation codes in that they are commands required for the execution of each. For example, $\overline{A}$ gates **8–22** through **8–26** are responsive to combinations of **U0–2** in order to respectively generate the output signals **0X**, **1X–2**, **2X–2**, **3X–2**, and **4X** via respective N gates **8–27** through **8–31**. As may be seen from an examination of $\overline{A}$**8–22**, inputs thereto are supplied by the clear output terminals of **U0–2** such that a significant negative output from $\overline{A}$**8–22** appears only when the higher order octal digit of the operation code is 0. The letter X associated with this output signal indicates that the lower order octal digit may be of any value from 0 to 7. In similar fashion, $\overline{A}$**8–23** is responsive to an output from the set terminal of **U02** and the clear terminals of **U0** and **U1** so that a negative output appears therefrom only if a higher order octal digit of 1 is preset in the operation code. A similar conversion into an octal digit representation is made upon the three binary bits held in **U3** through **U5** by $\overline{A}$ gates **A–32** through **A–38**. These $\overline{A}$ gates respectively generate signals **X0** through **X6** via N gates **A–39** through **A–45**. Thus, if **U3** and **U4** are each set to binary 1 while **U5** is cleared to binary 0, $\overline{A}$**8–38** detects the coincidence of positive signals to all of its inputs to generate a significant negative output which indicates that the lower order octal digit of the operation code has a value of 6.

Each of $\overline{A}$ gates **8–46** through **8–69** has an input derived from a respective one of the octal translators associated with the two groups of stages **U0–2** and **U3–5**. For example, $\overline{A}$**8–46** has an input from **N8–31** which itself generates the **4X** signal, with another input derived from **N8–39** generating the **X0** signal. Therefore, upon occurrence of positive signals on both of its input terminals, a significant negative output is generated from $\overline{A}$**8–46** which indicates that Op Code **40** is contained in the Op Code portion of the U register. Others of the above mentioned set of output $\overline{A}$ gates also generate signals indicating the presence of instructions **26**, **36**, **46**, etc. In addition, $\overline{A}$ gate **8–70** through **8–73** respectively generate multiple operation code signals such as **0X**, $\overline{\mathbf{X0}}$, **1X**, **U03**, etc. These code signals like the other multiple operation code signals **2X**, **3X**, etc. are designed to represent composite operation codes related to one another by at least one common command. Their use will be discussed subsequently when describing the command translator circuits of the U register.

Circuitry is also provided in FIGURE **8** to detect the presence of illegal operation code signals in the U register. As noted in Table 2, only thirty-five unique instructions are provided in the system, whereas 66 unique combinations are available (00 to 77) using a two digit octal representation. Therefore, illegal octal operation codes are those represented in Table 3 below.

TABLE 3

Signal:	Illegal Operation Codes
5X	50, 51, 52, 53, 54, 55, 56, 57
6X	60, 61, 62, 63, 64, 65, 66, 67
7X	70, 71, 72, 73, 74, 75, 76, 77
X7	07, 17, 27, 37, 47, 57, 67, 77

If an illegal operation code is detected in U0–5, then the Illegal Operation Code flip-flop **8–74** is set which will alter the operation of the program system. FF8–74 is set by a positive signal generated from $\overline{O}$**8–75** via **N8–76**. $\overline{O}$**8–75** in turn is responsive to $\overline{A}$**8–79** which senses the presence of 1 bits in all of the stages **U3** to **U5**, signifying the presence of an octal digit 7 in the lower order of the op code. Another input to $\overline{O}$**8–75** is derived from $\overline{A}$**8–77** which has one input from the set terminal of **U0** which, if set, indicates the presence of a 1 bit having the value 4 when considering the value of the higher order octal digit. The other input to $\overline{A}$**8–77** is derived from $\overline{O}$**8–78** which has a positive output therefrom if either U1 or U2 is set. Thus, $\overline{O}$**8–75** determines the presence of either a higher order octal digit of value 5, 6, or 7, or a lower order octal digit of value 7 or both. The setting of FF8–74 enables $\overline{A}$**8–81** and $\overline{A}$**8–82** at times MP12 and MP14, respectively. These signals first transfer **P*** to P so that the same memory address, from which the illegal instruction was derived, will again be referenced. The U register stages **0–9** are cleared (via $\overline{O}$40–19), so that the system cannot execute the illegal operation code. In addition, the signal Illegal Operation Code Error is generated from **N8–83**.

FIGURE 9 discloses the command translator circuits associated with U register stages **0–5** which generate the commands needed for execution of any particular instruction. The majority of the command signals for an instruction are present at the output of the command translator circuit as long as the Op Code remains in the U register stages **0–5**. Others, however, are only present during the time that a MP signal is applied to the circuit. For example, the negative signal representing instruction 45 from FIGURE 8 is applied via **N9–53** to **N9–7** and $\overline{A}$**9–10**. Instruction 45 calls for an input output data transfer as noted in Table 2 and at time MP15, $\overline{A}$**9–10** generates a command Set R Memory Add FF. The output from **N9–53** is also inverted by **N9–11** to provide a negative command signal indicating that the instruction is an instruction 46. In like fashion, instructions 22, 21 and 20 are inverted by respective N gates **9–54** through **9–56** and gated through respective $\overline{A}$ gates **9–12** through **9–14** at MP7 time. Thus, if the U register contains an instruction 22, the input to **N9–54** is negative which applied a positive input to $\overline{A}$**9–12** and results in a significant negative output therefrom upon occurrence of a positive MP7 pulse. The command generated from $\overline{A}$**9–12** is termed F2 To O which causes the transfer of information from the F2 register into the O register for storage in the variable memory.

An example of an untimed signal is from from $\overline{O}$**9–21** which derives inputs from both instruction 30 and instruction 33. Therefore, the command Load Constant/Increment R from $\overline{O}$**9–21** is present for the complete time that the instruction code is held in the U register. $\overline{O}$**9–22** is responsive to the presence of one of several instruction codes to generate a command U+R To U* Enable which is positive for the duration of the stored U register op code. $\overline{A}$**9–50** is responsive to the instruction 25 signal from FIGURE 8, as well as to a signal A Operand Reference Enable from FIGURE 39, to generate a significant negative output which in turn generates a command Clear X Enable. The signal Clear X Enable is also generated for an instruction 43 from $\overline{O}$**9–45** and **N9–46**, as well as for any of the instructions having an octal digit of value 1 in the higher order which is represented by the input signal 1X from FIGURE 8. Likewise, Clear X Enable command is generated for any instruction code having a higher order octal digit of value 0, excepting instruction 00. Thus, referring to Table 2, it is noted that the command Clear X enable is generated for any of the instruction codes 01 through 16, since said instruction codes utilize the X register in the arithmetic unit during an arithmetic operation on data.

Other commands of interest generated in FIGURE 9 are those from $\overline{O}$**9–44**, $\overline{O}$**9–42**, and $\overline{O}$**9–33**. From $\overline{O}$**9–44**, the command Operand Instruction allows a memory reference at the address in U***10–23** for those instructions requiring a data word operand during execution. However, execution of some instructions may not require an operand in the permanent memory. $\overline{O}$**9–43** senses the class of instructions where a W operand address is illegal and generates such a signal via **N9–43** and **N9–41**. $\overline{O}$**9–33** generates a signal Illegal Repeat whenever the instruction contained in the U register is a non-repeatable instruction. A non-repeatable instruction is one which should not follow the Repeat instruction 40. $\overline{O}$**9–36** and $\overline{O}$**9–34** thereby detect the presence of any of the following instructions which are classed as the non-repeatable instructions: 14, 15, 16, 25, 26, 30, 31, 32, 33, 34, 35, 36, 41, 42, and 46. Although not detected by $\overline{O}$ gates **9–34** and **9–36**, the Repeat Instruction 40 is itself considered to be non-repeatable.

The generation of the remaining commands required for execution of the instruction words is considered to be obvious in view of the above description. It may here be stated that operation codes are translated and commands generated in both the U register and the U* register in a manner subsequently to become clear in the operation of the invention.

FIGURE 10 of the drawings shows the U register stages **10** through **23** which provide storage for 14 bits of the instruction word that normally specify the unindexed memory operand "S" address. Addresses can be placed in U10–23 either manually from the computer control panel (by means not shown) or by transmission from the variable or permanent memory transfer registers O or Z, respectively. U10–23 may also be set by other circuitry subsequently to be described.

As previously described, the 14 bit address portion of an instruction word may be considered as comprising 5 octal digits specifying a word location in one of several places within the system. The binary bits comprising each of these five octal digits are contained in the following U register stages. U21–23 holds the lowest order octal digit, while **U10** and **U11** hold the two bits comprising the highest order octal digit which can have a maximum value of 3. U12–14, 15–17, and 18020 hold binary bits respectively comprising the remaining three octal digits of the address. Thus, if all of the **U10–23** register stages are cleared with the exception of **U12**, the memory address is 04000 which is the first location in the variable memory.

$\overline{A}$ gates **10–10** through **10–23** are conditioned by the signal Z To U0–23 from FIGURE 39 in order to transfer the contents of the Z register into **U10–23**. In like fashion $\overline{A}$ gates **10–25** through **10–38** sample the contents of the O register and gate same into U10–23 upon occurrence of the signal O To U0–23 from FIGURE 9. The transfer from either the Z or O registers into **U10–23** occurs at the same time that the operation code of the instruction word is transferred to U0–5. If an interrupt Op Code 46 is forced into the U register, then U register stages **11**, **19**, and **20** may be selectively set from FIGURE 43 so as to force one of seven different addresses into **U10–23** in accordance with the particular type of interrupt sub-routine which is to be performed. For example, if an error interrupt is requested by the computer, then **U10–23** is cleared and only stage **U11** subsequently set from FIGURE 43 so as to force the address 10000 into the address stages of the U register. On the other hand, if a Channel 1 input-output Interrupt is requested, then stages **U11**, U19, and U20 are all set so that the address 10030 is forced into **U10–23**. One other additional way to set U10–23 is by a signal appearing from $\overline{A}$**10–62** during MP13 time of the execution cycle of the Repeat Instruction 40 which sets all of the stages **U10–23** to a binary 1 content. The number thus held in **U10–23** represents the decimal value −1 which is used to recalculate the address of the Repeat Instruction by subtracting 1 from the the address held in P register. In any of the above means whereby U register stages **10–13** may be set, a Clear U10–**23** is first supplied from FIGURE 40 via **N10–40**.

If the octal address 03000 (indicating that the operand is held in the Accumulator register) is placed into U10–23, this is sensed by $\overline{\text{A}}$**10–41** which generates the signal A Operand Address via N–**42**. Inasmuch as the two higher order digits of 0 and 3 are unique to the A address, it is only necessary for $\overline{\text{A}}$**10–41**, to sample U register stages **10**, **11**, and **12**. Thus, if **U10** and **U11** are both clear, the highest order octal digit must be equal to 0. Furthermore, if **U12** is cleared, then the second highest order octal digit can have value no higher than 3, since the U12 stage represents a 4 value for this octal order.

The execution of a Selective Sense instruction 41 is also controlled by the U**10–23** portion of the U register. Generally speaking, this instruction enables a computer stop or a skip of the next instruction to be selected from the computer control panel. The **U10–16** portion of the instruction is used to select seven selective stops. For example, assume that the instruction acquired from memory and placed into the U register has an Op Code of octal value 41 and an address portion having a binary 1 value only in its **U10** position. If a negative signal Select Stop 1 from the control panel is applied to **N10–52**, the positive output therefrom is applied to $\overline{\text{A}}$**10–43** which receives as well the positive set output from **U10**. The command translator circuits of FIGURE 9 generate Select Sense Instruction from U**0–5** which applies a positive signal from **N10–47** to $\overline{\text{A}}$**10–43**. The simultaneous occurrence of positive inputs causes this $\overline{\text{A}}$ gate to generate a significant negative output signal which, when applied to $\overline{\text{O}}$**10–48** and via **N10–49** and $\overline{\text{O}}$**10–50**, causes the signal Select Stop to be generated. Thus, the main program of the machine is stopped because of the match between the control panel signed Select Stop **1** and the presence of a binary 1 in **U10**. On the other hand, if only **U11** were set instead of **U10**, no stop would be made. A similar control panel signal may be applied via one of N gates **10–53** through **10–58** in order to determine the presence, if any, of a binary 1 bit in one of the associated register stages **U11** through **U16**. $\overline{\text{A}}$ gates **10–44** through **10–53** provide the comparison means for these selective sense stop tests.

An equivalent type circuitry is used for the selective sense skip test, with the exception that if the skip condition is satisfied, the next instruction following the selective sense instruction is not executed rather than stopping the machine as in the case of a selective sense stop test. For example, one of the register stages U17–U23 of a selective sense instruction may contain a binary bit which, if it compares against a Select Skip 1–7 generated from the control panel, will cause a Select Skip signal to be generated from $\overline{\text{A}}$**10–61**. For example, if a Select Skip 6 signal from the control panel is applied to one input of $\overline{\text{A}}$**10–58** when U22 is set, then a signal is applied to $\overline{\text{O}}$**10–60** which in turn generates a positive signal to one input of $\overline{\text{A}}$**10–61**. The other input to $\overline{\text{A}}$**10–61** is supplied from U CTC of FIGURE 9 in the form of a Select Sense signal indicating that the operation code held in U**0–5** is 41.

The U* register stages **0–5** receive the Op Code portion of the instruction held by U0–5 at MP4 time of the machine cycle immediately following the machine cycle in which the instruction is placed in the U register. Thus, U***0–5** is filled in the cycle during which the instruction is executed, with some exception. $\overline{\text{A}}$ gates **11–10** through **11–15** sample the clear terminals of the U0–5 register stages, while $\overline{\text{A}}$ gates **11–16** to **11–21** sample the set terminals of the U0–5 stages. Therefore, there is no need to provide a separate clear pulse to the U***0–5** stages since double gating between the U and U* registers is performed. The operation code is also translated in the U***0–5** stages in a manner similar to the translation occurring in the U register. For example, the multiple operation signals **0X***, **1X***, **2X***, **3X** and **4X*** are respectively generated by a plurality of $\overline{\text{A}}$ gates **11–22** through **11–26** via N gates **11–41** through **11–45**. In similar fashion, the signals **X0***, **X1***, **X2***, **X3***, and **X4***, **X5***, and **X6*** are generated by a plurality of $\overline{\text{A}}$ gates **11–27** through **1–33**. These signals are thereafter combined in a plurality of $\overline{\text{A}}$ gates **11–46** through **11–72** in order to generate a single signal representing the instruction Op Code.

FIGURE 12 shows the command translator circuits for U***0–5**. The output signals from FIGURE 11 are utilized by a series of gates in order to generate certain commands necessary for the execution of an instruction contained in U*. Because of the previous detailed description of the U register command translator circuits in FIGURE 9, and because certain of the commands in FIGURE 12 will be explained later in conjunction with the operation of the invention, no detailed explanation is given of the development of the commands from the signals supplied from the U* register stages **0–5**. However, it should be noted that the commands developed in FIGURE 12 are for the most part different from the commands developed in FIGURE 9, or the same command in both figures may be generated at different times. Furthermore, between MP4 and MP9 times of the execution cycle, both U0–5 and U***0–5** contain the Op Code of the instruction being executed. Thus, the provision of two individual command translator circuits allows the development of commands peculiar to the acquisition or execution cycle of an instruction, as well as commands which may be necessary in both cycles.

The U***10–23** register stages shown in FIGURE 13 provide 14 bit storage for both the "T" modified memory address portion of an instruction word if indexing occurs, or the original "S" address portion if there is no indexing. Addresses can be placed in U***10–23** by unconditionally gating from **U10–23** at MP13 time (the "S" address), or from the U* carry tree of FIGURE 19 which is employed to add the address in **U10–23** register to a 14 bit number in R register and place the sum into U***10–23** (the "T" address). U***10** and U***11** hold the binary bits associated with the highest order octal digit of the address portion while U***21**, U***22**, and U***23** contain the binary bits associated with the lowest order octal digit. The remaining U* stages **12** through **20** are divided into three groups of 3 bits each which respectively hold the three remaining octal digits of the address.

Prior to entry of information into U***10–23**, these stages are cleared at MP12 time via **N13–10**. Subsequently, $\overline{\text{A}}$ gates **13–11** through **13–34** are enabled at MP13 time in order to transfer the address from **U10–23** into U***10–23**. This address may or may not be subsequently replaced by the modified "T" address during the next following machine cycle, but in any event, the referencing of memory for an operand data word is performed by using the address contained in U***10–23**.

As previously noted, many instructions require the "S" address portion of the instruction in U register be indexed by a number contained in the R memory, with the result being placed into U***10–23** for use in a memory reference operation. This indexing is generally performed by withdrawing a modifying address from the R memory in a manner subsequently to be described and placing same in the 14 bit R register stages **0–13**. Thereafter, at MP0 time of the execution cycle of the instruction, which occurs following the acquisition cycle, the address held in **U10–23** is added to the address in **R0–13** and this sum is placed into U***10–23**. FIGURE 13 shows a portion of the circuitry required to accomplish this addition, with FIGURE 19 showing the remaining addition circuitry. Only FIGURE 13 will now be described. For example, $\bar{A}$ gates **13–25** through **13–28** are used to place a 0 sum bit into their corresponding U* register stages **10–23** in the event that there is no carry of 1 from the adjacent lower order stage addition. Conversely, $\bar{A}$ gates **13–39** through **13–52** are employed to place a binary 1 sum bit into their associated U* stages **10–23** in the event that there is no carry 1 from the next lower order stage addition. Each of the $\bar{A}$ gates **13–53** through **13–66** samples two output signals from the U* CARRY TREE of FIGURE 19 indicating the equality or inequality of the binary bits held in the U and R register stages of corresponding binary order. For example, $\bar{A}$**13–66** has an input $\overline{U23 \neq R13}$ as well as an input $U23 \neq R13$ both generated from FIGURE 19. The bit held in stage **R13** (the lowest binary order position of the 14 bit modifying number) has the same binary order significance as the bit held in **U23**, inasmuch as the R register need only provide 14 bit storage for an address. Both of these input signals to $\bar{A}$**13–66** are positive in the event that the binary bit held in **U23** differs in value from the binary bit in **R13**. For example, if **U23** contains a binary 1 while **R13** contains a binary 0, then both signals $\overline{U23 \neq R13}$ and $U23 \neq R13$ will be positive. On the other hand, if both **U23** and **R13** contain a 0 bit, then the signal $\overline{U23 \neq R13}$ will be negative. Likewise, if **U23** and **R13** both contain a binary 1 bit, then the signal $U23 \neq R13$ will be negative. In similar manner, each of the $\bar{A}$ gates **13–53** to **13–65** are responsive to similar signals which indicates the equality or non-equality of the binary order bits held in corresponding U and R register stages. The output from $\bar{A}$ gates **13–53** through **13–66** are applied directly to $\bar{A}$ gates **13–25** through **13–38**, and are inverted via respective N gates **13–67** through **13–80** for application to $\bar{A}$ gates **13–39** through **13–52**.

The U* Add Tree of FIGURE 19 also generates a carry signal for each of the U* register stages **10–22**. This carry signal indicates whether or not the summation of bits in the adjacent lower binary order will result in a carry of 1 to the next order. These carry signals are applied and inverted via N gates **13–81** through **13–109** to $\bar{A}$ gates **13–25** through **13–38** and $\bar{A}$ gates **13–39** through **13–52**. If no carry of 0 is generated to the next higher order, then the signal appearing on its associated conductor from FIGURE 19 is negative such that the positive output from the associated one of N gates **13–81** through **13–109** will be positive. This conditions the associated $\bar{A}$ gates to respond only to the status of corresponding U and R bits from $\bar{A}$ gates **13–53** through **13–66**. However, if a carry 1 is present, then the signal is positive and conditions a first set of $\bar{A}$ gates **13–110** to **13–122** and a second set of $\bar{A}$ gates **13–124** through **13–137**, which respectively set or clear the U* stages **10–23**. A positive signal appearing on a carry lead from FIGURE 19 also provides a negative output from its associated one of N gates **13–81** through **13–109** thus disabling $\bar{A}$ gates **13–25** through **13–38** and **13–39** through **13–51**. However, for indexing to occur, all of the above four groups of $\bar{A}$ gates must be conditioned by U+R to U* Enable from FIGURE 9 which is applied via **N13–94** and $\bar{A}$**13–95** at MP0.

Table 4 illustrates a typical operation wherein the address in **U10–23** is added to the number in R and the result placed in U*.

TABLE 4

Stage	...12	13	14.....21	22	23
U......	0	1	0 1	0	1
R......	0	1	1 1	0	0
Carry..	1	1	1 0	0	0
U*.....	1	1	0 0	0	1

Initially assuming that **U23** contains a binary 1 while **R13** contains a binary 0, $\bar{A}$**13–66** is enabled by two positive inputs to develop a significant negative output which, when inverted by **N13–80**, allows $\bar{A}$**13–52** to generate a set U***23** signal at MP0 time. The negative signal from $\bar{A}$**13–66** disables $\bar{A}$**13–38** from providing a clear signal to U***23** at this time. No carry to the next higher order position will be developed by this addition. Next assuming that **U22** contains a binary 0 while **R12** contains a binary 0, the signal $\overline{U22 \neq R12}$ is negative such that $\bar{A}$**13–65** continues to provide a positive output therefrom to both $\bar{A}$ gates **13–37** and $\bar{A}$**13–122**. Furthermore, the signal Carry To U***22** from FIGURE 19 will also be negative such that a positive signal is applied via **N13–93** to $\bar{A}$ gates **13–37** and **13–51**. However, at gating time MP0, only $\bar{A}$ gate **13–37** has all of its inputs positive such that only it generates a significant negative output to clear U***22** and thus place binary 0 therein. No carry from this addition will be developed. In similar fashion, assuming that **U21** contains a binary 1 while **R11** contains a binary 1, the signal $U21 \neq R11$ is negative such that $\bar{A}$**13–64** is not enabled and thus continues to supply a positive signal to $\bar{A}$**13–36**. The signal Carry To U***21** is also negative which thereby allows only $\bar{A}$**13–36** to be enabled at the gating time to clear U***21** and thus set a binary 0 sum bit therein. A carry 1 results from this addition of two binary 1 bits and must be applied to the next higher order position of the adder which, although not specifically shown in FIGURE 13, has circuitry like that shown for stages **21** through **23**, with the exception that it derives inputs related to the binary bits in **U20** and **R10**. Next assuming that **U14** contains a binary 0 while **R4** contains a binary 1, and further assuming that the addition of the binary bits contained in **U15** and **R5** results in a carry 1, both input signals to $\bar{A}$**13–37** are positive because of the non-equality of the binary bits in the corresponding U and R stages. Therefore, $\bar{A}$**13–57** supplies a negative output to disable $\bar{A}$ gates **13–39** and $\bar{A}$**13–114**. At the same time a positive signal via **N13–71** is applied to $\bar{A}$**13–128** and $\bar{A}$**13–43**. The signal Carry To U*14 is also positive and applied to $\bar{A}$**13–128** and $\bar{A}$**13–114**. At MP0 time, only $\bar{A}$**13–128** has all inputs positive so that it provides a negative clear signal to the U***14** stage, thereby entering binary 0 therein. Now assuming that both **U13** and **R3** contain binary 1's, together with an input carry 1 from the adjacent lower order addition, it is seen that the signal $U13 \neq R3$ is negative so that $\bar{A}$**13–56** provides a positive signal to $\bar{A}$ gates **13–28** and **13–113**. In addition, the signal Carry To U***13** is positive which also causes $\bar{A}$**13–113** to be enabled at gating time to set a binary 1 into U***13**.

The operation of addition in the remaining U* stages is believed to be obvious in view of the description given above. Further clarification of the operation will be given during the subsequent explanation of the U* ADD TREE of FIGURE 19.

Also shown in connection with U***10**, **11**, and **12** are additional logic circuits for generating signals used in the memory access control circuits of FIGURE 39. $\bar{A}$ gates **13–144** and **13–145** are enabled by the same signals which enable $\bar{A}$ gates **13–25** and **13–26**, respectively, except for the conditioning signal applied via **N13–150**. $\bar{A}$ **13–146** and **13–147** are likewise enabled by the same signals which respectively enable $\bar{A}$**13–124** and $\bar{A}$**13–125**. The outputs from $\bar{A}$**13–144** and **13–146** applied to $\bar{O}$**13–138** which in turn is also supplied with an input from $\bar{A}$**13–141**. Likewise, outputs from $\bar{A}$**13–145** and $\bar{A}$**13–147** are applied to $\bar{O}$**13–139** which in turn has also input from $\bar{A}$**13–142**. As seen in FIGURE 13, the presence of the negative signal U+R To U* Enable conditions $\bar{A}$ gates **13–144** through **13–147** to generate signals indicative of the two highest order binary bits of the sum of R and U registers.

However, these signals from $\overline{\text{O}}$**13–138** and $\overline{\text{O}}$**13–139** appear by MP13 time of the cycle as soon as R register is filled with the index number and so are generated prior to the actual placement of the sum into U***10–23** at MP0 time. Where indexing is not permitted, the signal U+R To U* Enable is positive, which allows $\overline{\text{A}}$**13–141** and $\overline{\text{A}}$**14–142** to respectively sample the contents of U***10** and U***11**. These stages will not be subsequently changed at MP0 time.

$\overline{\text{A}}$ gates **13–149** and **13–148** are associated with U***12** in order to generate an early signal indicating the condition of U***12** subsequent to the indexing operation. The outputs of these $\overline{\text{A}}$ gates are directed to $\overline{\text{O}}$**13–140**, which in turn is also supplied with an input from $\overline{\text{A}}$**13–143** which samples the condition of U***12** in the event that no indexing is permitted. Therefore, these additional gates associated with U***12** perform essentially the same function as the above described additional gates associated with U***10** and U***11**.

FIGURE 14 shows **U** register stages **6–9** and U* stages **6–9** of the instruction register. These stages provide 4 bit storage for the **R** memory reference address designator "*j*" comprising a portion of the instruction word. **U6–9** are filled with binary bits at the same time that the other stages of the U register are filled. For example, $\overline{\text{A}}$ gates **14–10** through **14–12** are enabled by the signal Z To **U0–23** from FIGURE 39 to transfer the contents of the permanent memory Z register stages **6–9** into **U6–9**. In like fashion, $\overline{\text{A}}$ gates **14–14** through **14–17** are provided to gate the contents of the variable memory O register into U register upon occurrence of the signal O To **U0–23** from FIGURE 39. A 4 bit number may also be forced into **U6–9** during the execution of an Input Output Data Transfer instruction or an interrupt instruction in a manner similar to the forced setting of other stages of the U register previously described. For example, during execution of an Input-Output Data Transfer For Channel **1**, a binary number 1000 (decimal value 8) is set into **U6–9** in order to reference R memory address location 8. This is performed by a negative signal Set **U06** applied to **U6** which places a binary 1 therein, with the stages **7**, **8**, and **9** remaining clear. During an Input Output Channel **2** operation, the decimal value 10 is forced into **U6–9** by setting only **U6** and **U8** (1010), FIGURE 43. During execution of an Interrupt instruction, the decimal value 15 (1111) is forced into **U6–9** for an Error Interrupt, while the decimal value 14 (1110) is forced into **U6–9** for all other interrupts. The value 15 is stored by setting all of the stages **U6–U9**, while the value 14 is stored by setting only stages **U6**, **U7**, and **U8**.

There is provision made for gating the contents of **U6–9** into the corresponding stages U*609 under certain conditions. Thus, $\overline{\text{A}}$ gates **14–20** through **14–23** are provided to sample the clear output terminals of **U6–9** upon occurrence of a positive gating signal generated from $\overline{\text{O}}$**14–28**. In like fashion, $\overline{\text{A}}$ gates **14–24** through **14–27** sample the set output terminals of **U6–9** at the same gating time. Thus, a particular clear or set condition of a U register stage is double gated into the corresponding U* register stage. The conditions under which this transfer from U to U* may be determined by examination of the inputs to $\overline{\text{O}}$**14–28**. One such input is derived from $\overline{\text{A}}$**14–29** which in turn is enabled at MP12 time of a machine cycle. To provide a significant negative output from $\overline{\text{A}}$**14–29**, it is also necessary to provide positive inputs via **N14–30** and **N14–31**, which in turn require negative inputs of the respective signals Arithmetic Lock-out and No First R Memory Reference. If an arithmetic lockout has occurred, or if it is not required to perform an index operation upon the address contained in the **U10–23** then one of these signals is positive and thus prevents the transfer from **U6–9** to U***6–9**. An exception to this, however, occurs if the instruction Op Code in **U0–5** requires a Load R/Store R function such that $\overline{\text{A}}$**14–32** is enabled at MP4 time in order to provide a significant negative input to $\overline{\text{O}}$**14–28**.

If U***6–9** stages have been set by an R_j address, it may be necessary at times to clear these stages of their information to prevent the referencing of R memory. These clear signals are provided from $\overline{\text{O}}$**14–36** via **N14–37** to the clear input terminals of U***6–9**. The conditions under which a clear signal is generated from $\overline{\text{O}}$**14–36** may be noted by examining $\overline{\text{A}}$**14–38**, $\overline{\text{A}}$**14–39**, and $\overline{\text{A}}$**14–40**. Thus, assuming that an Arithmetic Lockout or a No First R Memory Reference condition is present, significant negative outputs will be provided from respective $\overline{\text{A}}$ gates **14–38** or **14–39** to $\overline{\text{O}}$**14–36**. Furthermore, U***6–9** is unconditionally cleared from $\overline{\text{A}}$**14–40** at MP4 time of a cycle unless the instruction executed is either a Load R/Store R, or if the system is operating in the repeat mode wherein the instruction contained in the instruction register is being repeated a number of times according to a value held in a specific R memory location. Also, the enabling of $\overline{\text{A}}$**14–40** is further conditional upon the absence of an input-output data transfer operation. When the system is operating in its repeat mode $\overline{\text{A}}$**14–33** is instead enabled at MP4 time of each cycle to force a decimal 7 into U***6–9** by setting stages **7** through **9** and clearing stage **6**. This R memory location is that holding the value specifying the number of times that the instruction contained in the instruction register is to be repeated. During an input-output data transfer operation, wherein a word is transferred either to or from the variable memory, $\overline{\text{A}}$**14–34** is enabled at MP4 time to set U***9** and thus change the value previously set into U***6–9** from the **U6–9** stages. The setting of U***9** thus adds a value of 1 to the previous value, so that the resulting value in U***6–9** refers to an R memory location wherein is located a variable memory address containing the last data word of a group to be transferred in the input-output operation.

3.2. *R Memory*

The 4 bit address placed into U***6–9** is used to reference a particular address location in the R memory for purposes of withdrawing a 14 bit number therefrom and placing same in the R register. Translation circuits are provided in FIGURE 14 and FIGURE 15 for converting the 4 bit binary number contained in U***6–9** into its decimal equivalent which is thereafter used by the R memory addressing circuits. For example, $\overline{\text{A}}$ gates **14–43** through **14–46** examine the contents of U***6** and U***7** to generate a group of signals at both MP5 and MP13 time which are used in the translator circuits of FIGURE 15. In similar fashion, $\overline{\text{A}}$ gates **14–51** through **14–54** examine the same stages U***6** and U***7** to generate a similar group of signals both at MP2 and MP10 times. The signals generated by each of these two sets of $\overline{\text{A}}$ gates have corresponding logical significance except that they are generated at different times during the machine cycle. $\overline{\text{A}}$ gates **14–59** through **14–62** examine the contents of U***8** and U***9**, but there is no gating signal required for outputs to be generated therefrom. In addition, the clear condition of U***8** results in a signal Transfer **C1** Enable, while the set condition of U***8** creates a signal Transfer **C2** Enable. These signals are subsequently described in connection with the input-output operation.

FIGURE 15 shows the remaining translation circuits for U***6–9**. Here the signals developed in FIGURE 14 are further combined in order to develop a single read signal and a single write signal, at different times, for referencing a particular one of the fifteen address locations of the R memory. For example, if U* stages **6**, **7**, and **8** have binary **0** bits stored therein, while the U***9** has a binary **1** bit, the decimal equivalent of this 4 bit address is equal to 1. In this event, the signal $\overline{\text{U*06}}$, $\overline{\text{U*07}}$–1 is positive and applied to $\overline{\text{A}}$**15–10**. Furthermore, the signal $\overline{\mathbf{U*08}}$, **U*09** is positive which is also applied to $\overline{\mathbf{A}}$**15–10**. Since the $\overline{\mathbf{U*06}}$, $\overline{\mathbf{U*07}}$**–1** signal is generated at times 5 and 13 during the machine cycle, it is seen that a significant output from $\overline{\mathbf{A}}$**15–10** is also generated at this time, which is used to energize the read driver **1** of the R memory so as to reference address location 1 therein and withdraw the 14 bit number to be placed in the R register. Since the R memory in the present embodiment is comprised of magnetic cores, this read-out is destructive. Therefore, a write operation follows the read operation to store the 14 bit number back into the address location 1 so that it will be available if future referencing of this location takes place. Also, new information may be written into the R Memory if desired. The write function is performed in the present system by $\overline{\mathbf{A}}$**15–11** which is responsive to the signal $\overline{\mathbf{U*08}}$, $\overline{\mathbf{U*09}}$ as well as to the signal $\overline{\mathbf{U*06}}$, $\overline{\mathbf{U*07}}$**–2** occurring at MP times 2 and 10 immediately following the respective read times of 5 and 13. The output from $\overline{\mathbf{A}}$**15–11** is applied to energize the word write driver **1**.

The selection of other address locations in the R memory is therefore believed to be evident in view of the preceding discussion.

FIGURE 16 of the drawings discloses details of the R register into which a 14 bit binary number is placed from the R memory when referenced. In addition, gating means **16–10** through **16–23** are provided to gate the contents of the P register into R under certain conditions. This latter function is performed by $\overline{\mathbf{O}}$**16–24** which in turn has two inputs either of which causes the signal P To R to be generated. One of these inputs is derived from $\overline{\mathbf{A}}$**16–25** which is at MP0 time and further requires a positive input from $\overline{\mathbf{O}}$**16–26**. The latter $\overline{\mathrm{O}}$ gate provides a significant positive output if either Interrupt Instruction or a Sub-routine Jump instruction is being performed, since for each it is necessary to store the address of the next instruction of the main program to be acquired from memory upon termination of the sub-routine. An input to $\overline{\mathbf{O}}$**16–24** is also provided from $\overline{\mathbf{A}}$**16–29** which in turn generates a significant negative output only at MP13 time of the Repeat Instruction 40 execution.

Prior to entry of information into R stages **0–13** is cleared by an output signal from $\overline{\mathbf{O}}$**16–30** via **N16–31**. R register is unconditionally cleared at MP4 and MP12 times via respjective N gates **16–32** and **16–33**. Also, $\overline{\mathbf{A}}$**16–34** and **16–35** provide clear signals for a Sub-routine Jump Instruction or an Interrupt Instruction in preparation for transfer of the P register into R register. However, in connection with $\overline{\mathbf{A}}$**16–34**, no clear signal for the R register is generated if the sub-routine jump instruction is to be skipped.

Prior to the transfer of information from R memory location to the R register, the R_j To R flip-flop **16–36** must be set to enable transmission of the R memory sense amplifiers to the set terminals of the respective R register stages **0–13**. The setting of flip-flop **16–36** is unconditional at MP5 time and also at MP13 time if a repeat instruction is not being executed, in which case $\overline{\mathbf{A}}$**16–38** will be disabled.

During execution of a repeated instruction, it is necessary to determine when the 14 bit value held in the R register is equal to —1. In the binary fractional arithmetic employed in the present system, this value is indicated by the presence of 1 bits in all of the R stages **0–13**. The circuitry provided in FIGURE 16 to detect this condition of the R register comprises $\overline{\mathrm{A}}$ gates **16–42** through **16–48**. For example, $\overline{\mathbf{A}}$**16–42** is responsive to 1 bits contained in all of the stages **R0–3** to generate a significant negative output therefrom which, when inverted via **N16–43** is applied to $\overline{\mathbf{A}}$**16–44**. Likewise, $\overline{\mathbf{A}}$**16–45** generates a significant negative output in response to 1 bits contained in all of the register stages **R4–8**, while $\overline{\mathbf{A}}$**16–47** responds to 1 bits in register stages **R9** through **R13**. The concurrence of outputs from $\overline{\mathbf{A}}$**16–42, 16–45**, and **16–47** is detected by $\overline{\mathbf{A}}$**16–44** which generates an output indicating that R=—1.

FIGURE 17 of the drawings shows details of the R memory counter which enables the contents held in the R register to be selectively incremented by the value 1 during certain operations of the program system. The R memory counter accepts inputs from the R register, and the output therefrom is either the original R register contents unmodified, or the R register contents incremented by 1. The output from the R memory counter is transferred via the OR gates to the R memory at the selected address location therein. If the R Memory Add flip-flop **17–52** is in its clear condition, then the number contained in the R register is transferred through the R memory counter unchanged. However, if the R Memory flip-flop **17–52** is set, then 1 is added to the lowest binary order bit of the R register, which is that residing in R13. Flip-flop **17–52** is unconditionally cleared from FIGURE 18 at MP4 and MP12 time. During an Input Output Data Transfer operation, however, it is set from FIGURE 9 in order to enable the system to determine how many words are to be transferred between the computer and its peripheral equipment. The flip-flop is also set during the execution of a repeated instruction.

As before indicated, either the original or incremented R number appears from $\overline{\mathrm{O}}$ gates **17–10** through **17–23** and is applied to the OR gates next to be described. These $\overline{\mathrm{O}}$ gates **17–10** through **17–23** in turn receive respective inputs from a set of $\overline{\mathrm{A}}$ gates **17–24** through **17–37** and from another set of $\overline{\mathrm{A}}$ gates **17–38** through **17–51**. $\overline{\mathrm{A}}$ gates **17–24** through **17–37** have respective inputs from the clear output terminals of R register stages **0–13**, while $\overline{\mathrm{A}}$ gates **17–38** through **17–51** have respective inputs from the set terminals of these R register stages. The other inputs to $\overline{\mathrm{A}}$ gates in these respective two sets are supplied from a set of $\overline{\mathrm{O}}$ gates **17–54** through **17–66** both directly and via a set of N gates **17–68** through **17–81**. Thus, a positive output from a respective one of the $\overline{\mathrm{O}}$ gates **17–54** through **17–66** enables the corresponding $\overline{\mathrm{A}}$ gate in the set **17–38** through **17–51**, and so a positive output from a respective one of $\overline{\mathrm{O}}$ gates **17–10** through **17–23** requires the presence of a binary 1 in the associated one of R stages **0–13**. Conversely, the presence of a negative output from the $\overline{\mathrm{O}}$ gates **17–54** through **17–66** causes the set of $\overline{\mathrm{A}}$ gates **17–24** through **17–37** to be responsive to the clear terminal of the associated R register stage. Thus, $\overline{\mathrm{O}}$ gates **17–54** through **17–66** determine the value of the carry supplied to the next higher order counter stage. If a carry 1 is supplied to a particular binary order of the counter, then the value of the output bit transmitted to the $\overline{\mathrm{O}}$ gate is the inverse of the bit residing in the corresponding R register stage. Conversely, if a carry 0 is transmitted to the counter stage, then the output bit has the same value as that residing in the corresponding R register stage.

The determination of the carry to each of the counter stages is accomplished in the following manner. Assuming that R Memory Add flip-flop **17–52** is initially cleared, a positive output is supplied to $\overline{\mathbf{A}}$**17–51** while a negative output via **N17–81** is directed to $\overline{\mathbf{A}}$**17–57**. Thus, counter stage **13**, which is responsive to the lowest order binary bit in R stage **13**, transmits a significant signal from $\overline{\mathbf{O}}$**17–23** only if the R stage contains a 1 bit. If the R stage **13** contains a 0 bit, then no output is derived from $\overline{\mathbf{O}}$**17–23** thus indicating a binary 0 value which is the same as that held in R13. Since FF17–52 is in its clear position, the output from its set terminal is negative so as to maintain $\overline{\mathrm{O}}$ gates **17–64, 17–65** and **17–66** in a condition whereby each generates a positive signal regardless of the value of the other inputs applied thereto. Therefore, the output bits appearing from the corresponding $\overline{\mathrm{O}}$ gates **17–20** through **17–22** correspond to the value of the bits residing in **R10** through **R12**. If $\overline{O}$ gate **17–64** has such a positive output, this is inverted by **N17–67** and a negative signal applied to all of the remaining $\overline{O}$ gates **17–54** through **17–63**. Thus positive signals are generated by each of these $\overline{O}$ gates which results in the transfer of the original bits in R stages **0–9** to the output of the R memory counter.

When **FF17–52** is set, **1** will be added to the bit contained in **R13** and carries generated as required to the remaining higher order stages of the counter. The clear output terminal from **FF17–52** is now negative which thus disables $\overline{A}$**17–51** and enables $\overline{A}$**17–37** via **N17–81**. The value of the output bit appearing from $\overline{O}$**17–23** will now be the inverse of the bit appearing in **R13** since, if the latter contains a binary 0, signal $\overline{\mathbf{R13}}$ is positive and thus generates a significant negative output from $\overline{A}$**17–37** which in turn energizes $\overline{O}$**17–23**. If, however, **R13** contains a binary 1, the positive value of the signal from **R13** set terminal cannot pass through $\overline{A}$**17–51** so that a negative signal is obtained from $\overline{O}$**17–23**. A positive signal from the set output of **FF17–52** is also applied to $\overline{O}$ gates **17–66**. The presence of a significant output from any of these gates depends upon the value of the bits residing in the R register stages. Thus, $\overline{O}$ gate **17–66** is responsive to a positive signal from **R13** if such contains a 1 bit, so as to generate a significant negative output which in turn disables $\overline{A}$**17–50** in order that the output from $\overline{O}$**17–22** will be the inverse of the bit residing in **R12**. This is correct, inasmuch as the addition of 1 to a 1 bit contained in **R13** results in a carry of 1 which is added to the bit in **R12** and changes its binary value. $\overline{O}$**17–65** in turn is responsive both to the positive output from **FF17–52** as well as to the bits contained in **R13** and **R12**. Therefore, the function of $\overline{O}$**17–65** is to determine if a carry 1 transmitted to counter stage **12** will be absorbed therein by virtue of a 0 bit being contained in **R12**. If not, then a carry 1 will be generated from counter stage **12** to counter stage **11** which will, when added to the bit in **R11**, change its value. The operation of $\overline{O}$**17–64** is similar in that it determines whether or not a carry 1 applied to counter stage **12** will be absorbed by the presence of the 0 bit in either **R11** or **R12**, and if not, disables $\overline{A}$**17–28** to provide an output from $\overline{O}$**17–20** which is opposite to the value of the bit residing in R.

The remaining $\overline{O}$ gates **17–54** through **17–63** operate in similar fashion in that each determines whether a carry 1 introduced into the first stage of a group of counter stages will be absorbed by virtue of there being a 0 in at least one of said group of stages. This operation is analogous to the operation of the address counter for the P register and consequently will not be described in further detail.

FIGURE 18 shows the OR gates used to accept information from various units and store same within a location of R memory as determined by the address contained in **U*6–9**. As mentioned before, the OR gates can accept information from the R register via the R address counter for storage within the R memory. This is accomplished by a set of $\overline{A}$ gates **18–38** through **18–51** which respectively sample the output of the R address counter stages R_j-Counter **0** through R_j-Counter **13**. A set of $\overline{O}$ gates **18–10** through **18–23** receives information from the above identified $\overline{A}$ gates and transmits same via a set of respective $\overline{A}$ gates **18–24** through **18–37** to the write circuitry of the R memory. However, in order to enable $\overline{A}$ gate **18–38** through **18–51**, the R To OR Gates flip-flop **18–93** must be set. This function is performed for four different conditions, which are as follows. In the event that neither a Load Constant or Increment R instruction is being performed, or if the system is not in its repeat mode, then a set signal is supplied from $\overline{A}$**18–94** at MP1 time. This is the operation normally occurring during execution of instructions. If, however, the system is operating in its repeat mode, then the setting of flip-flop **18–93** at MP1 time is inhibited and instead a set signal is applied from $\overline{A}$**18–95** at MP9 time, the purpose of which will subsequently be described. Likewise, a set signal from $\overline{A}$**18–96** is received at MP9 time if the instruction to be executed is either a Store R or an Input Output Data Transfer. Flip-flop **18–93** is unconditionally cleared at MP4 and MP12 times via $\overline{O}$**18–107**.

In addition to information from the R register, the R memory may be loaded from any of three other sources. These sources are **U*10–23**, the O register, and the X register found in the arithmetic section of the computer. For example, in the instruction Load R wherein a source other than the arithmetic unit is used to supply the data word, an output is received from $\overline{A}$**18–100** which sets flip-flop **18–99** and thus enables a set of $\overline{A}$ gates **18–52** through **18–65** to transfer information from the O register into the R memory at a particular address location. If, however, the Load R instruction requires information from the arithmetic unit to be stored within the reference memory, then an output signal is generated from $\overline{A}$**18–103** which sets flip-flop **18–102** and thus enables a set of $\overline{A}$ gates **18–66** through **18–78** to transfer a fourteen bit number from the X register. In similar fashion, an instruction may be contained in the U* register whose address portion **10–23** contains a constant which must be loaded into the reference memory. In such case, the Op Code of this instruction enables $\overline{A}$**18–106** at MP1 time to set flip-flop **18–104** to enable a set of $\overline{A}$ gates **18–79** through **18–92** to pass the 14 bit address portion from **U*–23** into the reference memory.

Having now discussed the U* register and the R register, the description of the U* carry tree in FIGURE 19 will be given. The signals $\overline{\mathbf{U10 \neq R00}}$, $\overline{\mathbf{U11 \neq R01}}$, etc., are generated from a set of $\overline{O}$ gates **19–10** through **19–23**. As before indicated, these signals represent the condition that binary 0 does not reside in both of the corresponding order R and U stages. For example, if both signals to $\overline{O}$**19–10** are positive, thus indicating that binary 0 resides in both the **R0** and **U10** stages, then the output from $\overline{O}$**19–10** is negative and indicates that the indicated nonequality is absent. However, if either one or both of the R and U stages contains a binary 1, then one or both of the input signals to $\overline{O}$**19–10** is negative, thus generating a positive signal therefrom. The signals **U10**$\neq$**R0**, **U11**$\neq$**R01**, etc. are generated by a set of $\overline{A}$ gates **19–24** through **19–37** respectively. As previous indicated, a positive signal from any one of these gates indicates that the indicated non-equality exists. For example, if one or both of the corresponding U and R stages contains a binary 0, then one or both input signals to $\overline{A}$**19–24** is negative, thus resulting in a positive output therefrom. If however, both **U10** and **R00** contain a binary 1, then both signals to $\overline{A}$**19–24** are positive which results in a negative output signal therefrom thus indicating that the indicated non-equality is absent.

The formation of the carry signal to each of the U* stages is somewhat more complicated. For example, the positive signal carry to **U*22** is generated whenever a binary 1 bit exists in both **U23** and **R13**, with this condition being sensed by $\overline{A}$**19–37** which thereby generates a significant negative output which in turn is inverted by **N19–50**. If this stage of the U* carry tree is denoted by numeral **13**, then it may be said that C_{13}, where this value is equal to 1, is equal to $R_{13} \cdot U_{23}$ expressed in Boolean algebra notation. Thus, if **U23** and **R13** each contains a binary **1**, then an output carry of value 1 is generated from stage **13** of the U* carry tree and is propagated to the input of **U*22**. A carry generated because of $R_{13} \cdot U_{23}$ can, for purposes of this discussion, be considered as a type C^1. In FIGURE 19, this carry will be generated by $\overline{A}$**19–37** which senses the absence of the non-equality $U23 \neq R13$. A generalized form of this first type of carry C_n^1 may be given as $R_n \cdot U_n$.

Upon addition of the binary bits contained in U22 and R12, a carry output therefrom may be generated in one of two different ways. The first type of carry C^1 is that generated when both of the binary bits added together in this order are binary 1's. In FIGURE 19, this C^1 type of carry is generated by Ā19–36 whose output is supplied to Ō19–49 for application to U*21. A second kind of carry 1 which may be generated is that resulting if either one of these binary bits is 1 and the incoming carry from the adjacent lower order (higher numbered stage) is a carry 1. Thus, in FIGURE 19, the incoming carry signal from the adjacent lower order stage of the U* carry tree is that appearing from N19–50 which is applied to Ā19–62. In addition, Ō19–22 detects the presence of a binary 1 bit in at least one of the U22 or R12 register stages so as to produce a positive output therefrom which is also applied to Ā19–62. The output carry from Ā19–62 is applied to Ō19–49 which again directs the carry to U*21. Thus, Ā19–62 generates a carry of type C_n^2 which may be calculated by $(R_n+U_n)C_{n+1}^1$, where C_{n+1}^1 is the first kind of carry generated in the adjacent lower order stage having, in the case of the U* carry tree, a numerical designation one higher than the stage under consideration. In turn, the carry C_{n+1}^1 is generated due to the presence of binary bits in both R_{n+1} and U_{n+1} such that $C_{n+1}^1=R_{n+1}\cdot U_{n+1}$.

When adding together the binary bits found in U21 and R11, three different types of carries C_{11}^1, C_{11}^2, or C_{11}^3 may be generated within this stage, each of which is transmitted to U*20. For example, a C^1 type of carry may be generated due to the presence of binary bits in both R11 and U21 as detected by Ā19–35 whose output is directed to Ō19–48. A C^2 type carry is generated due to the presence of a binary 1 in either U21 or R11 coupled with a carry 1 from the next adjacent lower order stage of the U* carry tree, which is that stage dealing with the addition of U22 and R12. Inasmuch as stage 12 of the U* carry tree generates a carry either of the C^1 type or the C^2 type, then either can be used to add to the binary 1 bit in stage 12 to thereby generate a carry to stage 20. For example, Ō19–21 detects the presence of a binary 1 bit in either U21 or R11 and this output is directed to Ā19–61 which in turn has an input thereto from Ā19–36 via N19–79. Since the output of Ā19–36 indicates that a carry of the type C^1 is generated within stage 12 of the U* carry tree, it is seen that Ā19–61 generates a carry of the type C^2, because

$$C_{11}^2=(R_{11}+U_{21})C_{12}^1$$

The output of Ō19–21 is also directed to Ā19–70 which has further inputs thereto from Ō19–22 and N19–50. Thus, Ā19–70 provides an output according to the equation $C_{11}^3=(R_{11}+U_{21})(R_{12}+U_{22})R_{13}\cdot U_{23}$. However, it may be noted that the value $(R_{12}+U_{22})R_{13}\cdot U_{23}$ actually equals C_{12}^2. Therefore, it is seen that stage 11 of the U* carry tree generates carries of value 1 formulated in one of three different ways and designated C_{11}^1, C_{11}^2, and C_{11}^3, all of which are transmitted by Ō19–48 to the U*20 stage. Obviously, one or more significant inputs may be present to Ō19–48 during a particular add time, however, only one of these inputs is necessary in order to generate and transmit a carry to the next adjacent high order stage of the U* register.

In stage 10 of the U* carry tree, which is that stage dealing with the addition of the binary bits contained in U20 and R10, four different types of carries C_{10}^1, C_{10}^2, or C_{10}^4 may be generated therein for transmission to the U*9 register stage. For example, a carry of the C^1 type may be generated from Ā19–34 which detects the presence of $R10\cdot U20$. A carry of the C^2 type is generated from Ā19–60 by detection of the presence of a binary 1 in either one of the U20 and R10 stages, coupled with detection of a carry C_{11}^1 from stage 11 of the U* carry tree. Since C_{11}^1 is generated whenever a binary bit 1 exists in both R11 and U21, the outputs form these two register stages may be applied to Ā19–60 so as to complete the necessary inputs for generation of the C_{10}^2 type of carry. A C_{10}^3 type of carry is generated by Ā19–69 which again receives an output from Ō19–20 as well as outputs from Ō19–21 R12, and U22. Since Ō19–21 detects the logical function $R11+U21$, it may be seen that Ā19–69 is a C^3 type of carry generated by the equation $(R10+U20)C_{11}^2$. A fourth type of carry C_{10}^4 is generated within carry tree stage 10 by detecting the presence of a 1 bit in either R10 or U20 plus the input of a C_{11}^3 type carry from the carry tree stage 11. Thus, Ā19–75 is responsive to Ō19–20 together with other inputs which are logically combined according to the following equation.

$$C_{10}^4=(R10+U20)(R11+U21)(R12+U22)(R13\cdot U23)$$

In this equation, the second term is applied by Ō19–21, the third term by Ō19–22, and the fourth term by R13 and U23. As will be evidenced by an examination of this equation, the last three terms actually equal the equation for the C_{11}^3 carry. Thus, when the adjacent lower order stage generates three different kinds of output carries, then one of four different types of carries with value 1 may be generated by the stage in question.

To recapitulate the above, Table 5 shows the formation of each of the four different kinds of carries C_n^1, C_n^2, C_n^3 and C_n^4 which may be applied to the next higher order stage.

TABLE 5

$$C_n^1=R_n\cdot U_n$$
$$C_n^2=(R_n+U_n)(R_{n+1}\cdot U_{n+1})=(R_n+U_n)C_{n+1}^1$$
$$C_n^3=(R_n+U_n)(R_{n+1}+U_{n+1})(R_{n+2}\cdot U_{n+2})=(R_n+U_n)C_{n+1}^2$$
$$C_n^4=(R_n+U_n)(R_{n+1}+U_{n+1})(R_{n+2}+U_{n+2})(R_{n+3}\cdot U_{n+3})=(R_n+U_n)C_{n+1}^3$$

These equations may be utilized in understanding the carry formation within the remaining U* carry tree stages of FIGURE 19. For example, Ā gates 19–25 19–37 provide carries of the type C_n^1. Ā gates 19–51 through 19–62 provide carries of the type C_n^2. Ā gates 19–63 through 10–70 provide carries of the type C_n^3. Ā gates 19–71 through 19–75 provide carries of the type C_n^4. Ō gates 19–38 through 19–49 collect the different types of carries generated within their respective associated stages and transmit same to the adjacent higher order U* register stage for use therein as has previously been described. Since the stages (excluding the first and last) in FIGURE 19 are divided into groups of three, an output group carry to the lowest order stage of the next higher group is considered as a C^1 type carry to that stage. Therefore, only four types of carries need be defined.

FIGURE 20 of the drawing shows the comparison circuits for determining the equality, if any, between the 14 bit number held in R register stages 0 through 13 and a 14 bit number held in U* register stages 10–23. The positive signal $R=U^*$ is generated from Ā20–10 via N20–11, and is utilized for control functions later to be described. In order to provide positive signals to all of the inputs of Ā20–10, significant negative outputs from all of the Ā gates 20–12 through 20–15 must be provided, which are respectively inverted by N gates 20–16 through 20–19. Each of the Ā gates 20–12 through 20–15 in turn must be provided with positive signals to all of its inputs. This is accomplished only when each bit in the U* register stages 10–23 corresponds in value to the bit held in the corresponding binary order register stages R0 through R13. Thus, Ā gates 20 and 20–21 respectively sample the clear and set output terminals of the corresponding register stages U*10 and R0. If a binary 0 is contained in both, then $\overline{A}$**20–20** generates a significant negative signal which is inverted via $\overline{O}$**20–22** and applied to one input of $\overline{A}$**20–12**. Conversely, if both **U*10** and **R0** contain binary 1 bits, then $\overline{A}$**20–21** is energized to again supply a positive signal via $\overline{O}$**20–22**. $\overline{A}$**20–23** examines the clear output terminals of **U*11** and **R01** to determine the presence of binary 0 values in both of these stages, while $\overline{A}$**20–24** examines the set output terminals of each of these stages to determine the presence, if any, of binary 1's in both. In similar fashion, $\overline{A}$ gates **20–26** and **20–27** look for equality of binary values in the **U*12** and **R02** in order to provide a significant positive input to $\overline{A}$**20–13**. The operation of the remaining logical gates in FIGURE 20 is therefore believed to be obvious. The comparison R=U* is required for the input output data transfer instruction to determine when the Input Output Data Transfer operation should be terminated.

3.3. *Variable Memory*

FIGURE 21 shows details of the V register which is the address register associated with the variable memory and to which is transferred a number specifying a location within said memory. As shown, the V register contains stages **V0** through **V10** which provides storage for only 11 binary bits. Although 14 bit addresses are normally employed, it is noted that the address range for the variable memory is from 04000 through 07777, expressed in octal fashion. Therefore, it is always known that a variable memory address must have binary 0 bits contained in the two highest order binary positions of the address, as well as a binary 1 bit contained in the third highest binary order position. Consequently, there is no need to provide storage for the three highest order binary positions of the 14 bit address when such address is sent from P or U* registers to the variable memory V register. This may be seen by noting that $\overline{A}$ gates **21–10** through **21–20** gate the contents of P register stages **3–13** to **V0–10**. Therefore, P register stages **0**, **1**, and **2** are not transferred to the V register since their contents are already known. In like fashion, $\overline{A}$ gates **21–22** through **21–32** gate the contents of **U*13–23** into the V register, thereby neglecting to sample **U*10–12**. The V register is unconditionally cleared at **MP0** time of each machine cycle.

With an address stored in V register, means must be provided to translate the bits contained therein so as to select a unique location in the variable memory, which in the present embodiment is comprised of magnetic cores arranged in a three dimensional array. While many different forms of translators may be employed for purposes of this invention, the specific one employed in the present system is shown in FIGURE 21 and is comprised of the following circuits. $\overline{A}$ gates **21–35** and **21–36** are responsive to the positive signal from the clear output terminal **V0** to generate signals at the memory read and write times, respectively. $\overline{A}$ gates **21–37** and **21–38** likewise are responsive to the set output terminal of **V0** to generate signals respectively occurring at the read and write times of the memory cycle. Outputs from N gates **21–39** through **21–42** are in turn applied to a series of $\overline{A}$ gates **21–43** through **21–50** in the manner shown, where they are combined with respective clear and set output signals from **V1** in order to generate a single signal at both the read and write times signifying the binary bit combination held in **V0** and **V1**. Thus, if **V0** is cleared while **V1** is set, then $\overline{A}$**21–44** generates a signal $\overline{\textbf{V0}}$**, V01** at the read of the memory cycle, while $\overline{A}$**21–48** generates the signal $\overline{\textbf{V0}}$**V1** during the write time of the memory cycle. These signals are utilized in the access switches of FIGURE 3 to enable the final selection of the X and Y coordinate drive lines used for reading and writing out of a memory address location. This operation will be more fully described in subsequent paragraphs.

$\overline{A}$ gates **21–51** through **21–58** sample and logically combine the contents of **V2**, **V3**, and **V4** in order to generate a single signal indicating said combination. In like fashion, $\overline{A}$ gates **21–83** through **21–90** logically combine the contents of **V8** through **V10** to generate a single signal unique to a particular binary combination held therein.

$\overline{A}$ gates **21–59** through **21–62** are responsive to respective clear and set outputs from **V5**, as well as to signals occurring from FIGURE 23 at the read and write memory times, to generate further signals which are logically combined with the signals from **V6** and **V7** in $\overline{A}$ gates **21–67** through **21–82**. Because of the read and write time signals associated with these and $\overline{A}$ gates, there will be one output signal emitted therefrom at the read memory time and a corresponding signal emitted therefrom at the memory write time. This operation is believed to be obvious in view of the similar operation occurring for $\overline{A}$ gates **21–43** through **21–50**.

FIGURE 22 shows details of the O transfer register associated with the variable memory and into which can be placed a 25 bit word (including parity) either acquired from memory during the read time or which must be written into memory during the write cycle. The O register can be loaded from a variety of locations. For example, a 24 bit word from X register may be gated to **O0–23** via $\overline{A}$ gates **22–10** through **22–23** upon generation of a command X To O from FIGURE 12. During an Input Data Transfer operation utilizing channel **2**, the contents of **C2** register may be gated into O register stages **0–24** via $\overline{A}$ gates **22–34** through **22–58**. In the case of an Input Data Transfer operation, a parity bit accompanies the 24 bit data such that the parity bit **O24** may be set at this time. The contents of E register may be gated into **O0–23** via $\overline{A}$ gates **22–59** through **22–82** by a signal from the E register. Thus, these three transfers of words from X, C2, and E are performed when it is desired to store their contents into variable memory at an address held by the V register.

O register in addition may be filled by information from a variable memory address location during its read cycle. The variable memory sense output amplifiers are connected to the set input terminals of **O0–24** so that upon detection of a binary 1 bit stored in a selected magnetic core, a negative signal from the sense amplifier sets the corresponding O register stage. O register is unconditionally cleared at **MP0** time.

Fourteen bit numbers from other locations in the system may also be gated into O register stages **10–23** (and sometimes **24**). In this category are the numbers stored in R, F1, F2, and E* registers. $\overline{A}$ gates **22–129** through **22–142** transfer a 14 bit number from F1 into **O10–23**, while $\overline{A}$ gates **22–44** through **22–157** gate the contents of F2 register into **O10–23**. These two operations are performed during the execution of a Store F1 or Store F2 instruction. Also, the contents of **E*10–23** must be stored in **O10–23** on certain occasions when an error has been detected in the system. This transfer is effected by $\overline{O}$ gates **22–84** through **22–96** directly from the designated E* register stages.

The set output terminals of the O register stages **10–24** are connected to a variety of locations within the system in order to transfer from the O register a number acquired from a variable memory address location. A set of $\overline{A}$ gates **22–67** through **22–191** are also provided which are connected to the clear output terminal of **O0–24** so that during the write time of a variable memory reference cycle, the inhibit winding of the memory matrix plane associated with the bit position of the O register stage may be energized to prevent a binary 1 from being written in the storage core by the coincident X and Y drive currents. This operation will be described further at a later time. Thus, if an O register stage is in the cleared condition (representing binary O) the positive signal from the clear output terminal switches with a Gate

Inhibit signal from FIGURE 23 during the memory write time to generate a significant positive output from one of the associated N gates **22–192** through **22–216**. Such a positive signal enables the associated inhibit generator to energize the inhibit winding. This technique is well known in the prior art.

The parity stage **24** of the O register may also be separately set by parity generating circuits as well as being set by the parity bit associated with an incoming data word. For example, words held in certain locations in the system do not have a parity bit associated therewith, such as the 14 bit numbers held in F1, F2, and R registers, as well as the 24 bit number held in the X register. It is necessary therefore to generate a parity bit for these numbers when they are to be stored in the variable memory. The generation of any of the gating signals F1 To O, F2 To O, R To O, or X To O causes the Insert Parity FF**22–160** to be set via $\overline{\text{O}}$**22–162**. The setting of FF**22–160** thereupon enables $\overline{\text{A}}$**22–65** to be responsive to the signal Even Parity Sets O24 from the Z and O parity check circuits, which perform the task of generating the parity for a number held in the O register. The calculated parity for a number, with which no parity is normally associated, is therefore placed into O24 for transfer with the O0–23 number into the variable memory. The signal E And E* To O also allows the setting of FF**22–160** in the same manner.

FIGURE 23 shows the variable memory control flip-flops which determine the read and write times when referencing the variable memory. If the variable memory is to be referenced for either an instruction or an operand, the flip-flop **23–10** is set by any one of the three signals **P** To V–1, P To V–2 or U* To V of FIGURE 39. The setting of **FF23–10** enables $\overline{\text{A}}$**23–15** and $\overline{\text{A}}$**22–26** to generate read and write signals, respectively, upon the unconditional setting of flip-flops **23–13** and **23–24** respectively. Thus, in a variable memory reference during a machine cycle, the reading of information from a particular address location in the variable memory occurs beginning at MP3 time while information is written into the variable memory at a time beginning MP10. The particular address location referenced is that indicated by the number held in the V register for the complete machine cycle. Writing of information into a core storage location is accomplished through use of an inhibit pulse which is used to determine the presence, if any, of a binary 0 digit in the associated O register stage. This signal is generated by flip-flop **23–23** which is set at MP9 time just prior to the setting of FF**23–24**. All flip-flops **23–10, 23–13, 23–21** and **23–24** are unconditionally cleared at specific times during the machine cycle, such as shown in FIGURE 23.

The Variable Memory itself is made up of 25 matrix planes, each containing 2048 magnetic cores of the bistable, square hysteresis loop type. One of these matrices **2–10** is shown in FIGURE 2 and is that which stores the binary bit of order significance 2^0 of each of the 2048 words in Variable Memory; e.g., the core **2–11** in the lower-left corner of the matrix stores bit 2^0 of that word stored at octal address 04000. On another 2048 core matrix plane identical to the one shown in FIGURE 2, the core in the lower-left corner stores bit 2^1 of the same word (at octal address 04000), and so forth for the remaining 23 planes. The X and Y drive lines that pass through bit 2^0 of octal address 04000 also pass through the remaining 24 bits (including the parity bit) associated with octal address 04000. Therefore, when an address is selected, all of the cores associated with the selected word are simultaneously sensed. Similarly, cores **2–12** in the upper-right corner of the matrices store those bits associated with the word at octal address 07777. Here again, all cores at octal address 07777 are at the same relative physical location on each matrix, and each core, from 2^0 through 2^{24} is located on a different 2048-core matrix plane.

There are four groups of eight Y drive lines each, although only two groups are shown in FIGURE 2 for clarity. The left end of each Y drive line in a group is connected to a different one of a set of eight Y Drive Circuits **Y0** through **Y7**, each of which in turn is connected to a different one of the $\overline{\text{A}}$ Gates **21–51** through **21–58**, respectively in the V register translator. The right ends of all Y drive lines in the same group are connected to the same access switch of a set of four Y Read/Write Access Switches **Y0** through **Y3**, each of which in turn has a Read terminal connected to a different one of $\overline{\text{A}}$ Gates **21–43** through **21–46**, respectively and a Write terminal connected to a different one of $\overline{\text{A}}$ Gates **21–47** through **21–50**. The right ends of other Y groups are connected to different ones of the Y access switches. Since only one Y Driver and one Y access switch can be enabled at any given time, only one Y drive line will be energized during a Variable Memory reference.

There are eight groups of eight X drive lines each, although only two groups are shown in FIGURE 2 for clarity. The upper end of each X drive line in a group is connected to a different one of a set of eight X Drive Circuits **X0** through **X7**, each of which in turn is connected to a different one of the $\overline{\text{A}}$ Gates **21–83** through **21–90**, respectively. The lower ends of all X lines in the same group are connected to the same access switch of a set of eight X Read/Write Access Switches **X0** through **X7**, each of which in turn has a Read terminal connected to a different one of $\overline{\text{A}}$ Gates **21–67** through **21–74**, respectively, and a Write terminal connected to a different one of $\overline{\text{A}}$ Gates **21–75** through **2–182**, respectively. Since only one X Driver and one X access switch can be enabled at any given time, only one X drive line will be energized during a Variable Memory reference.

When a **P** TO V or U* TO V pulse gates the address from the **P** or U* registers into the V register, the translators associated with V immediately enable those driver circuits associated with corresponding X and Y drive lines. The negative input to a driver allows it to respond to the subsequent enabling of the corresponding X and Y access switches when the Read Shaper FF**23–13** is set in order to force current in one direction through each of the selected X and Y drive lines. The effect of this coincident current in the X and Y drive lines is to switch all of the cores at the selected address to the "0" state. If a core was initially in the "0" state, a small signal is generated on the corresponding matrix plane sense winding threading the core (not shown in FIGURE 2), and a positive output signal is forwarded to the $\underline{0}$ register. However, if the core is initially in the "1" state, the direction of magnetization of the core reverses and a large signal is generated on the sense winding. This signal is amplified and forwarded to the gated detectors (not shown in FIGURE 2) where a strobe pulse from $\overline{\text{A}}$**23–11** gates a negative signal into the $\underline{0}$ register to set the associated flip-flop to the "1" state. The read cycle is initiated on MP3 when the Read Shaper flip-flop is set; the cycle ends on MP7 (delayed 0.05 microsecond) when the Read Shaper flip-flop is cleared. This technique is well known in the prior art.

Since the address in the V register remains the same during the complete memory cycle, the access switches and driver circuits that are enabled for the read cycle are used also for the write cycle. The selected driver circuits remain enabled throughout both operations. However, when the Read Shaper flip-flop is cleared, the X and Y access switches are disabled. When the Write Shaper flip-flop **23–24** is set, the write translators in the V register are gated and enable signals are issued to the Write terminals of the same X and Y access switches, so that current flows in the opposite direction through each of the selected X and Y drive lines. The write operation always follows the read operation since the latter destroys the information held in the word location. By the beginning of the write cycle, the cores at the selected address have all been cleared to the "0" state by prior read operation. The write currents of opposite direction attempt to switch all of the cores at the address location back to the "1" state. However, the data word in the Q register determines what is written into the selected address. For example, if the flip-flop **10** of the Q register is set to "1," the X and Y write drive currents switch the core at 2^{10} of the selected address to the "1" state. However, if the flip-flop **10** is cleared to the "0" state, an inhibit winding associated with the matrix plane containing this core (not shown in FIGURE 2) is energized by **A22–177** at the selected address which cancels the effect of the current in the X drive line. Since the Y drive line current acting alone cannot switch the core, the core remains in the "0" state. The write cycle is initiated on MP10 when the Write Shaper flip-flop is set; the cycle ends on MP14 (delayed 0.05 microsecond) when the Write Shaper flip-flop is cleared.

3.4. *Permanent Memory and Twister Input Switch*

FIGURE 24 shows the W register for holding an address of a location in the permanent memory. W register stages **0** through **13** provide 13 bit storage for a permanent memory address ranging from 10000 through 33777, and 37720 through 37777, with the latter range being the locations contained in the twister input switch which acts as a buffer between the critical input units and the computer. **Ā24–10** to **24–23** gate the contents of the P register into the W register stages upon occurrence of the signal P to W. In similar fashion, Ā gates **24–24** through **24–37** gate the contents of U* register stages **10–13** into the W register upon occurrence of the signal U* To W. The W register stages are unconditionally cleared at MP0 time via **N24–38.**

As with the V register, translators must be provided for the W register in order to select a particular location in the permanent memory or in the twister input switch. The W register translators are comprised of five groups. Ā gates **24–39** through **24–44** individually sense the binary combination held in **W0–2** in order to select one of five permanent memory D units or the twister input switch. As will further be described in connection with FIGURE 3, the permanent memory is divided into five units, **1** through **5**, in which different ones of the permanent memory address locations are found. Unit **1** contains addresses 10000 through 13777, unit **2** contains addresses 14000 through 17777, unit **3** contains addresses 20000 through 23777, unit **4** contains addresses 24000 through 27777, while unit **5** holds address locations 30000 through 33777. For example, if **W0–2** have binary bits 0, 1 and 0 respectively, then **Ā24–39** is enabled to select the permanent memory unit 1. This is so since W stages 0 and 1 hold the octal digit 1, while the cleared condition of W2 indicates that the maximum value of the second highest order octal digit is three. If **W0–2** contains 101, respectively, then **Ā24–42** is enabled to select the permanent memory unit 4, since **W0–1** contain the octal digit 2 while the set condition of **W2** indicates that the minimum value of the second highest order octal digit must be at least 4. It is therefore thought obvious as to the operation of the other unit selector Ā gates in view of the above discussion. In addition, **Ō24–45** is responsive to a significant negative output appearing from either Ā gates **24–39**, **24–41**, or **24–44** in order to generate a significant positive Unit Select Parity Check Signals.

The second group of the W register translators to be discussed are Ā gates **24–26** through **24–49** associated with **W4** and **W5** used to generate one of the Gate Line Access switch **X0–X3** signals therefrom. Although the clear and set output terminals of **W5** are connected directly to these associated Ā gates, the clear and set output terminals of **W4** are first applied to Ā gates **24–50** and **24–51** whose outputs are thereafter inverted via **N24–52** and **N24–53** before application to the Ā translators. The other input to Ā gates **24–50** and **24–51** is derived from FIGURE 27 when a Gate Group And Line X signal is produced in a manner subsequently to be described. The signals from Ā gates **24–46** through **24–49** are further applied to the permanent memory gating logic in FIGURE 25 before they are used in referencing the permanent memory.

The third group of translator circuits in FIGURE 24 are Ā gates **24–55** through **24–62** associated with W stages **6** through **8**. These Ā translators are responsive to binary bit combinations in **W6–8** in order to generate one of the Enable Group Access Switch **X0–X7** signals which is utilized as shown in FIGURE 3 subsequently to be described. In addition, the one outputs from **Ā24–61** or **24–62** is also used to reference the twistor input switch access circuits in a manner subsequently to be described. Register stage W6 must first be gated through Ā gates **24–63** and **24–64** by the Gate Group And Line X signal from FIGURE 27. Thus, the signals from **Ā24–46** through **24–49**, and **24–55** through **24–62** do not appear until a permanent memory read cycle has commenced. Ō gates **24–54** and **24–67** are also provided having inputs from the indicated Ā gate translators in order to perform a parity check operation. The fourth group of translators in FIGURE 24 are the Ā gates **24–68** through **24–75** associated with W register stages **3**, **9**, and **10** in order to generate one of the signals Gate Line Access switch **Y0–Y7.** These signals are directed to the permanent memory gating logic of FIGURE 25 where they are utilized in a manner subsequently to be described before application to the permanent memory access circuits of FIGURE 3. **W3** must first be gated through Ā gates **24–76** and **24–77** by the signal Gate Line Translators Y from FIGURE 27 before Ā gates **24–68** through **24–75** may generate a signal output therefrom.

The last group of W register translators are **Ā24–81** through **24–88** associated with **W11–13.** These Ā gates sense the binary combination held therein to generate one of the signals Enable Group Access Switch **Y0–Y7,** used both for permanent memory or a twistor input switch reference in a manner subsequently to be described. However, before such a signal can be generated the output of **W11** must be gated through Ā gates **24–89** and **24–90** by a Gate Group Translators Y signal from FIGURE 27. Ō gates **24–80** and **24–93** are respectively associated with indicated ones of the Gate Line Access Switch Y translators and Enable Group access switch Y translators in order to afford a parity check on the address held in the W register.

FIGURE 25 shows the permanent memory gating logic which accepts a Unit Select signal, a Gate Y line Access Switch signal, and a Gate X Line Access Switch signal from FIGURE 24 to generate time two signals used to reference the permanent memary or twistor input switch. For example, Ā gates **25–22** through **25–29** are responsive to the generated one of corresponding Gate Line Access Switch **Y0–Y7** signals, as well as to a generated Select Unit **1** signal, to generate one of the Enable Line Access switch **Y0–Y7** signals used to energize a Y drive line in permanent memory unit **1.** In similar fashion, Ā gates **25–30** through **25–33** respond to the generated one of the Gate Line Access Switch **X0–X3** signals from FIGURE 24, as well as the Select Unit **1** signal, to generate one of the Enable Line Access Switch **X0–X3** signals used to energize a X drive line in unit **1.** Similarly, a generated Select Unit **5** signal from FIGURE 24 is applied to Ā gates **25–73** through **25–84** which in turn are also respectively responsive to the generated one of the Gate Line Access Y switch signals and to the generated one of the Gate Line Access X switch signals in order to generate Y and X Enable Line Access signals for the permanent memory access circuits of unit **5.** Although not shown in FIGURE 25 for the sake of clarity, three other groups of $\overline{A}$ gates are also included in the permanent memory gating logic, each group being responsive to the Gate Line Access Switch Y and Gate Line Access Switch X signals from FIGURE 24, as well as to a different one of the Select Unit **2** through **4** signals. The X and Y Line Access output signals from each of these three groups are respectively applied to the permanent memory access circuits of units **2** through **4**.

$\overline{A}$ gates **25–87** through **25–90** are respectively responsive to a generated one of the Gate Line Access Switch signals Y7, Y6, Y5, or Y4 from FIGURE 24, as well as to the Select Twistor Input Switch signal. These $\overline{A}$ gate output signals are directed to access switch units **Y0** through **Y3** in the twistor input access circuits in order to enable reference of same if the address in the W register so requires. Furthermore, $\overline{A}$**28–86** responds to the Select Twistor Input switch signal as well as to the Gate Line Access Switch X3 signal to enable the single line access switch **X0** in the twistor input access circuits. The use of the signals generated in FIGURE 25 will become apparent during the discussion to follow with regard to FIGURE 3.

FIGURE 26 shows details of the Z transfer register which is provided to accept information from any of the five D units in permanent memory or from the twistor input switch. Each of the permanent memory units has a complete set of 25 sense amplifiers individual thereto, with each sense amplifier in a set being connected via $\overline{O}$ gates **26–60** through **26–84** or $\overline{O}$ gates **26–85** through **26–109** to the associated Z register stages **0–24**. The actual setting of the Z register stages, however, is not done until a Strobe Z signal is generated from FIGURE 27 which permits $\overline{A}$ gates **26–10** through **26–59** to enter the word being read. The Z register is unconditionally cleared at MP0 time of each machine cycle.

FIGURE 27 shows the control circuits for the permanent memory which include an X Shaper **FF27–10**, a Y Shaper **FF27–11**, a Permanent Memory Strobe **FF27–12** and a Twistor Input Strobe **FF27–13**. Timing pulses from the main pulse distributor initiate the control circuits to provide the sequence for a permanent memory reference. When the X shaper **FF27–10** is set at MP2 time, the output from the 1 side of the flip-flop gates both the X group and X line translators in the W register. At the same time, the output from the clear side of **FF27–10** causes $\overline{O}$**27–14** to gate the Y group translators in the W register. The effect that these signals have on the overall operation of the permanent memory will become evident in subsequent paragraphs.

When the Y Shaper flip-flop **27–11** is set at MP3 time, after a delay of 0.20 microsecond, its output gates the Y line translators in the W register. At the same time, the negative signal from the clear output terminal of **FF27–11** is also applied via $\overline{O}$**27–14** to insure the generation of the Y Gate Group Translator signal. When both the X and Y Shaper flip-flops **27–10** and **27–11** are thus set, a X drive line and a Y drive line will be energized within one of the permanent memory units or within the twistor input switch to apply coincident currents to a word access core enabling same to sense the contents of the selected address location. Since read out from the permanent memory or from the twistor input switch is non-destructive and since no information is ever transmitted from the computer therto, there is no provision for a write cycle.

The Strobe Enable for the sense amplifiers is provided by **FF27–12** when the permanent memory is referenced or by **FF27–13** when the TIS (twistor input switch) is referenced. If the permanent memory is referenced, then the Select Twistor Input signal from FIGURE 24 is positive to enable $\overline{A}$**27–18** to set **FF27–12**. On the other hand, if TIS is selected, then a positive signal is applied to $\overline{A}$**27–19** which sets **FF27–13** through a small delay. $\overline{O}$**27–15** is common to the clear output terminals of both flip-flops in order to generate a positive Strobe Z Sense Amplifiers signal which is forwarded to the Z register in order that these sense amplifier outputs can be examined to determine the contents of the address location.

A block diagram of the permanent memory access circuits is shown in FIGURE 3. The permanent memory is comprised of five identical independent D units **1** through **5**, of which the top memory plane of unit **1** is shown in the figure. Each unit contains four 512 word three dimensional matrix stacks, or a total of 2048 storage locations per unit. There are 26 memory planes in each stack. Each matrix stack **1**, **2**, **3**, and **4** of a unit has associated therewith 16 X drive lines and 32 Y drive lines, although for clarity many of the X and Y drive lines have been omitted in FIGURE 3. The medium of storage in the permanent memory is a group of magnetized program cards. Each 512 word matrix stack requires 16 cards, with each card being inserted vertically into the stack in parallel with one of the 16 X drive lines associated therewith. The end views of two such program cards **3–10** and **3–11** are shown in matrix stack **4**. A 32 by 26 array of ferrite "squares" is embedded in the surface of each card. Therefore, each magnet card stores 32 26-bit words, one bit for each of the memory planes in the stack. The permanent memory uses a 24 bit word plus one parity bit, with the 26th bit location being a spare. Only the ferrite squares at those bit locations that are to represent 0's are magnetized. All ferrite squares not magnetized represent binary 1's.

The presence or absence of a "magnetized" ferrite square is sensed by a wire wrapped with magnetic tape placed close to the square. This wire is called a twistor. A twistor wire is formed by wrapping a piece of magnetic permalloy tape around a copper wire at a 45 degree angle. The magnetic tape detects information at particular bit locations, and the copper conductor transmits the information to a sense amplifier. An individual group of 26 twistor wires is provided for each unit, one for each memory plane, which are common to all four matrix stacks. A solenoid strip for each word location in a stack is looped around all 26 of the twistor wires and one end of the loop is linked with a word access core, several of which are shown in FIGURE 3 and designated **2–12**. An X drive line and a Y drive line links each of the word access cores of which there are 512 for each stack. There are thirty-two solenoid strips associated with each magnet card which is inserted in the stack adjacent thereto. Therefore, the embedded ferrite squares are placed directly over the twistor wires. For further details of the twistor storage element, reference may be had to an article by A. H. Bobeck, "A New Storage Element Suitable for Large-Sized Memory Arrays—The Twistor," appearing in The Bell System Technical Journal, vol. 36, No. 6, pp. 1319–1339 (November 1957).

For each permanent memory unit, there are four groups of eight X drive lines each, although only two such groups are actually shown in FIGURE 3 for clarity. The left end of each X drive line in a group is connected to a different one of a set of eight X Group Access Switches and Inverters **X0** through **X7**, each of which in turn is connected to a different one of $\overline{A}$ gates **24–55** through **24–62**, respectively, in the W translator. There is a set of X Group Access Switches for each of the memory units, with the output from each of the said $\overline{A}$ gates being connected to the correspondingly numbered switch in each set. The right ends of all X drive lines in the same group are connected to the same Line Access Switch of a set of four X line Access Switches **X0** through **X3**, each of which in turn is connected to a different one of $\overline{A}$ gates **25–30** through **25–33** for unit **1**. X drive lines of other groups in unit **1** are connected to other of the switches **X0** through **X3**. The right ends of the X drive lines in other units are connected to ones of a different set of X line access switches for that particular unit, which in turn are connected to other $\overline{A}$ gates in FIGURE 25 some of which have been omitted for the sake of clarity. For example, in unit **5**, the right ends of all X drive lines in the same group are connected to the same access switch of a different set of four X line access switches X**0** through X**3**, each of which in turn is connected to a different one of Ā gates **25–81** through **25–84**.

For each permanent memory unit, there are eight groups of eight Y drive lines each, although only two such groups are shown in FIGURE 3. The upper end of each Y drive line in a group is connected to a different one of a set of eight Y Group Access Switches and Inverters Y**0** through Y**7**, each of which in turn is connected to a different one of Ā gates **24–81** through **24–88**, respectively. As with the X Group Access Switches, a set of Y Group Access Switches is provided for each of the permanent memory units, with the output of each of the above identified Ā gates being connected to the same correspondingly numbered Y switch in each set. The lower ends of all Y lines in the same group are connected to the same Line Access Switch of a set of eight Y Line Access Switches Y**0** through Y**7**, each of which in turn is connected to a different one of a set of Ā gates **25–22** through **25–29** in the case of unit **1**. For other units, such as unit **5**, a different set of Y Line Access Switches is provided which in turn are respectively connected to Ā gates **25–73** through **25–80**. Therefore, from the discussion above, it is seen that only one X Group Access Switch and one X Line Access Switch can be enabled at any given time in only one of the permanent memory units, so that only one X drive line will be energized during a permanent memory reference. Also, only one Y drive line in only one of the memory units will be enabled in a given time. If TIS is selected, however, then no permanent memory units will be referenced.

When the computer is turned on, a pre-magnetizing D.C. current is passed through each twistor wire before the permanent memory on TIS is put into operation. This current magnetizes all of the twistor wires uniformly and eliminates the stray fields that are induced in the twistor wires where the program cards are inserted in the slots. After an address has been placed in the W register, the X Shaper FF**27–10** is set to gate the W group translators to enable a Group Access Switch for an X and a Y drive line in all of the memory units. Furthermore, the setting of FF**27–10** gates the W line translators to enable a X line access switch in only one of the memory units. When the Y Shaper FF**27–11** is set, the line translators are gated to enable a Y line access switch in only one of the memory units which is that in which the X line access switch is also enabled. Therefore, there is energized X and Y drive lines through a single word access core at the location specified by the W register. The effect of the coincident X and Y currents is to reverse the direction of magnetic flux in the word access core and, as a result, induce a voltage in the solenoid strip linked to the core. The induced voltage in the solenoid strip sets up a magnetic field in the 26 twistor wires that overcomes the magnetic field produced by the D.C. current. Thus, the direction of magnetization of each twistor wire within the solenoid strip reverses, causing a voltage to be induced in the twistor wire. This induced voltage is transmitted along the wire to an associated sense amplifier (not shown in FIGURE 3). The change in magnetization in the twistor wire can be inhibited by placing a magnetized ferrite square over the solenoid strip and twistor wire in such a manner that the field produced by the solenoid strip is insufficient to overcome the external field of the permanent magnet. Thus, if a permanent magnet (a magnetized ferrite square) is present, the magnetization of the twistor wire is not changed, and a binary 0 is considered read from the bit location. Where there is no magnet, i.e., the ferrite square is not magnetized, the direction of magnetization of the twistor wire within the solenoid strip changes and a 1 is read from the bit location.

During the read cycle, the current in the X and Y drive lines reverses the direction of magnetic flux in the word access core at the selected address. Since the permanent memory has no write cycle, a bias winding (not shown in FIGURE 3) is used to change the core back to its original state. The bias winding passes through all of the word access cores and the current is applied to the winding continuously. The current in the bias winding opposes the current flow in the X and Y drive lines. Therefore, when the X Shaper flip-flop **27–10** is set, the X drive current overcomes the current in the bias winding. When the Y Shaper flip-flop **27–11** is set, the direction of magnetization in the word access core changes. When both of these flip-flops are cleared, the bias current automatically switches the word access core back to its original state.

There are sense amplifier circuits connected to the 25 twistor wires for each unit that detect the twistor bits that change, amplify the signals, and forward the resulting data to corresponding flip-flops in the Z register. Each twistor wire has its own sense amplifier. On MP8, the Permanent Memory Strobe FF**27–12** is set, thus causing strobe pulses to be issued to Ā gates **26–10** through **26–59** to gate binary 1 signals into the corresponding flip-flops in the Z register.

The twistor input switch is also a word organized storage device that receives data from critical input units and makes the data immediately available to the computer. Although not shown in FIGURE 3 in detail, the characteristics and principles of operation of the twistor input switch are similar to those of the permanent memory. However, since data from the input unit is changing continually, electro-magnets are used to effect the storage of data in contrast to the use of permanent magnets on the program cards in the permanent memory. Although the computer must be stopped in order to change the contents of the permanent memory, the contents of the twistor input switch is modified while the computer is operating whenever the solenoid coils are energized or de-energized. Signals from the critical input units determine the states of the solenoids. As in the permanent memory, a solenoid strip is looped around the 26 parallel twistor wires of the unit for each word location. The Y drive circuits in the twistor input switch are similar to those in the permanent memory and consist of eight Group Access Switches enabled by Ā gates **24–81** through **24–88**, and four Line Access Switches enabled by Ā gates **25–87** through **25–90**. Each Y Line Access Switch drives eight lines and each line, in turn, connects to an associated group access switch. The X drive circuits consist of two Group Access Switches respectively enabled by Ā gates **24–61** and **24–62**, and one Line Access Switch enabled by Ā**2586**. These circuits operate in the same manner as the circuits in the permanent memory. Except for the Strobe flip-flop **27–12**, the other control flip-flops **27–10** and **27–11** used with the permanent memory are also used with the twistor input switch.

3.5. *Channel 1 Input/Output*

FIGURE 28 shows details of the F1 register, which has thirteen stages F**1–00** through F**1–12** for holding a 13 bit number gated from U* register stages **10–22** via a set of Ā gates **28–10** through **28–22**. This number from the U* register is the T address portion of the instruction intiate F1 having an octal operation code of 31. Such an instruction is withdrawn from either the permanent or variable memory and its "S" address portion is indexed to place the sum into U* register stages **10–23**. Thereafter, the OP Code 31 causes transmission of the U* **10–22** to the F**1** register by generation of the command U* To F1 from FIGURE 9.

The function of the 13 bit number held in **F1–00** through F**1–12** is to control the operation of an output data transfer on channel **1**, which further includes a 25 bit (including parity) word transfer register **C1** which provides for buffering between the computer variable memory and an external unit. For example, the bits contained by **F1–00** and **F1–01** determine the direction of transfer of the data word between the computer and the unit. Therefore, these two bit positions of the F1 register may be considered to define the function of the input-output operation. In the present embodiment, channel **1** is only utilized for an output operation such that a data word is only transferred from the variable memory of the computer to one of several external units associated with channel **1**. If **F1–00** is cleared while **F1–01** is set after transfer of the number from the stages U*10 and U*11, respectively, then an output operation utilizing channel **1** is indicated. Thus, $\overline{A}$**28–41** detects the presence of bits **01** in **F1–00** and **F1–01**, respectively, to generate a signal via **N28–42** indicating that an output operation is to occur. If both **F1–00** and **F1–01** are set, then an Illegal Function combination is indicated which is sensed by $\overline{A}$**28–43** to generate an F1–Function Type error signal at commencement of the output operation.

F1 register stages **2** through **4**, not specifically shown in FIGURE 28, are not used in the channel **1** output operation and so their bit contents are therefore not significant. F1 register stages **5**, **6**, and **7** are utilized to select one of several different banks of output devices to which the data word from variable memory in the computer is to be transferred. For example, $\overline{A}$ gates **28–24** through **28–30**, respectively, sense the presence of one of seven unique bit combinations held in the F1–5 through F1–7 register stages to provide a signal enabling a particular one of several different output devices to prepare for receipt of the data word. At least one of the F1–5 through 7 stages must be set upon transfer of the U* register thereto, or else an F1-Group Designated error is generated by $\overline{A}$**28–24** upon commencement of the channel **1** output operation.

F1 register stages **8**, **9**, and **10**, not specifically shown in FIGURE 28, are used to select one particular external unit within the selected output group as determined by the setting of **F1** register stages **5** through **7**. Thus, if **F1–09** stage is set, the second output unit in the selected output group will be the one actually receiving the transfer data word. Register stages **F1–11** is not utilized in the channel **1** operation.

If a binary 1 bit from U*22 sets **F1–12**, this signifies that upon termination of the channel **1** output operation, an Input-Output Channel **1** Interrupt request is made such that an interrupt sub-routine will be initiated by the computer. If, on the other hand, **F1–12** remains clear during the channel **1** output operation, then no such interrupt request will be initiated. Both **F1–13** and **F1–14** are control flip-flops associated with the F1 register, with **F1–14** being set to initiate the channel **1** output operation under control of the number held in the **F1** register. The setting of **F1–14** is by $\overline{A}$**28–38** which is responsive at MP9 time to the presence of an Initiate F1 op code held in **U*0–5**. When once set, **F1–14** remains in this condition until the required number of data words have been transferred to the selected output unit, after which **F1–13** is set to subsequently clear **F1–14** by an output from $\overline{A}$**28–47**. The output from $\overline{A}$**28–38** utilized to set **F1–14** also generates an initial control signal F1–Prime Output from $\overline{A}$**28–40** whose use will subsequently be described. It is not until **F1–14** is set that gating signals will be applied to $\overline{A}$**28–43** and $\overline{A}$**28–44** to generate either a Function Type Error signal or a Group Designator Error signal, or both if such are to be found. These two signals are applied to $\overline{O}$**28–45** which in turn indicates a Channel **1** error and furthermore causes the immediate settings of **F1–13** to clear **F1–14** and immediately terminate the channel **1** output operation before the first data word can actually be transferred.

FIGURE 29 shows details of the C1 register which provides a 25 bit buffer storage (including parity) for the data word being transferred between the computer variable memory and the external unit selected by the value held in the F1 register. A 24 bit number from the O register, which is that output register associated with the variable memory, can be gated into C1 register via set of $\overline{A}$ gates **29–10** through **29–34** upon application thereto of a Gate O To C1 via **N29–35**. The output from the **C1** register is directed to all of the output units associated with channel **1** but is only actually received by the one selected bẏ the **F1** register number. Further details of channel **1** output units are not shown herein inasmuch as they do not comprise a portion of the present invention.

FIGURE 30 shows circuitry for controlling the channel output operation and providing synchronization between the computer and external unit selected. In order to initiate a channel output operation, the External Resume **FF30–10** must be set in one of two ways. Initially, this is done by output from $\overline{A}$**29–40** which is generated because of the presence of the Initiate F1 Op Code 31 in **U*0–5**. Otherwise, if there is more than one data word to be transferred under control of the same F1 number, then the output unit itself requests all subsequent data words. This latter function is indicated in FIGURE 30 by the signal C1–Resume which is normally negative unless the output unit requests that a data word be transferred to it. In addition to **FF30–10**, an External Resume lockout **FF30–12** is provided which is also set at commencement of the channel **1** output operation to clear External Resume FF**30–10** via $\overline{A}$**30–15** and subsequently lock out any further setting of it until the positive signal C1 resume is terminated from the output unit. The positive signal C1–Resume is terminated as soon as the operation has commenced to transfer a data word, and it will stay negative until the output unit has received the data word from the computer.

A C Status **FF30–18** is also provided which, when set, indicates that the C1 register has been filled with the data word from variable memory and is waiting for transfer to the selected output unit. In its cleared condition, **FF30–18** indicates that C1 register is empty. A Terminate **FF30–30** is set after the last data word to be transferred under control of the F1 register number has been withdrawn from variable memory and placed into the C1 register. The setting of **FF30–30** thereupon causes the channel **1** output operation to be terminated at the time that the output unit receives the last data word from C1 register and request another.

A brief operation of FIGURE 30 will now be described without detailed reference to other figures of this application. The acquisition of an Initiate F1 instruction from memory, which is ultimately placed in U* register, causes the External Resume **FF30–10** to be set by the signal F1 Prime Output from FIGURE 28. The set condition of **FF30–10**, coupled with the presence of a positive signal F1 Output And Go from the F1 register which indicates that an output operation of channel **1** is to be commenced, allows $\overline{A}$**30–11** to generate a signal at MP15 time which in turn sets **FF30–12** and clears **FF30–18**. In addition, the output from $\overline{A}$**30–11** is gated via $\overline{O}$**30–13** to clear C1 register and at the same time is applied via **N30–32** to $\overline{A}$**30–14** to request an input-output data transfer operation for channel **1**. Any output from $\overline{A}$**30–14** of course assumes that **FF30–30** is in its cleared condition at this time. The output from $\overline{A}$**30–14** is directed to the priority circuits of FIGURE 42 where it requests an interrupt of the main program to be made for one machine cycle. During this machine cycle, a data word is extracted from a particular location in variable memory whose address resides in a specific location in the R memory. The data word from variable memory is thereupon placed in the O register, whereupon a signal Channel **1** Ready is applied to set the C status **FF30–18**. This thereby indicates that C1 register is full, with Channel **1** Ready signal also simultaneously gating the contents of O into C1 register. Thereafter, the external unit can at its leisure extract the data word from C1 register, after which time it generates a positive C1–Resume which is clocked at any one of the MP times 1, 5, 9 and 13 to again set **FF30–10**. The setting of this flip-flop allows **Ā30–11** to generate a signal requesting an output data transfer in channel **1** for the second data word, if such is required. This process is repeated until the last data word to be transferred under control of the F1 register has been withdrawn from variable memory and placed in C1. At this time, a Terminate Channel **1** signal is generated from FIGURE 12 which sets **FF30–30** so that a subsequent request from the output unit for another data word will cause the generation of a signal from **Ā30–31**. This signal is directed to **F1–13** in FIGURE 28 which when set, subsequently clears **F1–14** and so terminates the output operation under control of the F1 register. Thereafter, a new Initiate F1 instruction must be acquired from memory and placed in U* register in order for an output operation on channel **1** to again be initiated.

3.6. *Channel 2 Input/Output*

FIGURE 31 shows details of the F2 register utilized in controlling the input-output data transfer operations in channel **2**. It provides 13 bit storage for a number gated in from **U*10–22** via a set of Ā gates **31–10** through **31–22** in a manner similar to the loading of the F1 register. The contents of **U*10–22** at this time is the T address portion of an Initiate F2 instruction which is withdrawn from memory during the course of the main program sequence in the same manner that an Initiate F1 instruction may be acquired. However, in the present embodiment, a data transfer may be made in either direction between the computer and the external units. Among the units associated with channel **2** are magnetic tape units and a typewriter, as well as other external units.

As with the F1 register, stages **F2–00** and **F2–01** indicate the direction of transfer in channel **2**. However, inasmuch as transfers may be made in both directions in this channel, both F2 Input and F2 Output signals are generated from these two stages. For example, if an output operation is to be performed, i.e., a data word taken from variable memory of the computer and transmitted to one of the output units, then **F2–01** is set and **F2–00** is clear such that **Ā31–41** will generate an F2 Output And Go signal upon initiation of the output operation. Conversely, **F2–00** is set and **F2–01** is clear, then **Ā31–52** will generate an F2 Input And Go signal upon initiation of the operation. As with channel **1**, **Ā31–43** senses the presence of **1** bits in both **F2–00** and **F2–01** to generate an F2 Function Type Error which is applied to **Ō31–45** to indicate a Channel **2** Error, as well as terminating the operation before it can actually accomplish the first data transfer. Furthermore, the presence of binary 0's in both **F2–00** and **F2–01** is detected by **Ā31–55** to cause a magnetic tape unit to back space or rewind without there being any transfer of a data word to or from this unit.

F2 register stages **2** through **4** are utilized for a variety of purposes when the selected output device is a magnetic tape unit. For example, if **F2–02** is set, then an even parity is given to the 6 bit character written on the magnetic tape. Otherwise, an odd parity is given to the character if a binary 0 stands in **F2–02**. The four unique combinations of the two binary bits held in **F2–03** and **F2–04** are utilized to either Backspace, Rewind, or Rewind With Interlock the magnetic tape unit selected.

F3 stages **5** through **7** select the particular group of units or to from which the data transfer is directed, in a manner similar to the group selection in the F1 register. Ā gates **31–24** through **31–30** thereupon are responsive to seven unique combinations of the three binary bits held in F2 stages **5** through **7** in order to generate one signal selecting the output group. However, if none of the F2 stages **5** through **7** is set after transfer of U* into the F2 register, then an F2–Group Designator Error signal will be generated from **Ā31–44** which is also applied to **Ō31–45** to indicate a Channel **2** Error and terminate the input-output operation.

F2 register stages **8** through **10** are utilized to select the particular external unit within the designated group. For example, in the present embodiment, a binary **1** in **F2–10**, with binary 0's in **F2–8** and **F2–9**, will select magnetic tape unit **1** within the group designated by an output signal from the set of Ā gates **31–24** through **31–30**. In the event that **F2–08** is set, this indicates that the typewriter is selected as the unit involved in the data transfer. Whenever a tape unit or the typewriter is involved, however, group **7** must always be selected which is indicated by a signal from **Ā31–30** due to the presence of binary 1's in all of the F2 stages **5** through **7**. Therefore, the signal from **Ā31–30** transmitted via **N31–37** enables **Ā31–56** or **Ā31–57** to indicate either a tape or a typewriter operation.

F2 stage **11** is used during a magnetic tape operation for a tape data sorting interrupt if a binary 1 is contained therein. If a typewriter output operation is being performed, then the presence of a binary 1 therein will inhibit a parity check. For output units other than tape or typewriter, the contents of **F2–11** are not significant.

F2–12, if set, indicates that a Channel **2** Input-Output Interrupt request must occur upon termination of the data transfer operation. The function of this stage is the same as the function of the corresponding stage **F1–12**. **F2–13** and **F2–14** likewise have the same functions as corresponding stages **F1–13** and **F1–14** previously described in connection with FIGURE 28. Thus, upon initiation of the data transfer operation, whereupon one or more data words may be successively transferred in channel **2** under control of the number residing in the F2 register, **F2–14** is set to generate the enabling signal F2–Go which is applied to various ones of the gates in both FIGURE 31 and others. The setting of **F2–13** is accomplished from any one of a variety of sources after the data transfer operation is terminated, i.e., after the required number of data words has been transferred to or from the unit. The setting of **F2–13** causes **Ā31–47** to clear **F2–14** and so terminate the F2–Go enabling signal. In like fashion, **Ā13–48** is responsive to a set condition of **F2–12** as well as to the set condition of **F2–13** to generate a request for an input-output channel **2** interrupt operation which is applied to the priority circuits of FIGURE 43. It is also possible to generate the Input-Output Channel **2** Interrupt request by a signal Interrupt Or Terminate from FIGURE 33 via **Ō31–49**.

FIGURE 32 shows details of the C2 buffer register which provides 25 bit storage (including parity) for the data word involved in the channel **2** input-output transfer operation. The data word may be transferred from the variable memory O register via a set of **Ā32–10** through **32–34** upon generation of a signal Gate O to C2. The number residing in C2 register from such transfer may then be utilized by the selected output unit which extracts the contents of C2 register at its leisure. Furthermore, the C2 register is utilized as the buffer for a channel **2** Input operation whereby a selected unit sends the data word to the computer to be stored in the variable memory. The C2 register may be loaded for an input operation to the computer from one of two different sources, depending upon the particular unit involved. For example, a twenty-five bit number may be gated in parallel fashion from an external unit via a set of Ā gates **32–35** through **32–61** upon generation of the signal Gate **DTU** To C2. However, if a magnetic tape unit of the typewriter is the unit involved, then the C2 register must successively be loaded with 6 bit characters from the D register stages **D00** through **D05**. Although more specifically shown in FIGURE 34, it may here be said that **D00** through **D05** is successively gated to C2–00 through C2–05, C2–06 through C2–11, C2–12 through C2–17 and C2–18 through C21–23. In other words, four successive transfers of 6 bits each must be accomplished in order to fill the C2 register with a twenty-four bit number supplied by a magnetic tape unit or a typewriter. After the C2 register has been completely filled, its content is gated to the O register from whence it is written into the variable memory.

FIGURE 33 is the C2 sequence control unit which provides synchronization between the computer and the input-output units associated with channel 2. For an output data transfer, much of the circuitry in FIGURE 33 is exactly similar to that provided in FIGURE 30 associated with channel 1. For example, the External Resume FF33–10 is set upon request by the unit (other than tape or typewriter) for a data word from the computer, or by the presence of an Initiate F2 Op Code 32 in the U* register stages 0–5. Ā33–11 is responsive to the set condition of FF33–10, as well as to a signal from F2 register indicating an output operation, to set the External Resume Lockout FF33–12, clear the C Status FF33–21, clear C2 register, and request an Input-Output Data Transfer Interrupt operation for channel 2 via Ō33–14 and Ā33–15. After generation of the signal from Ō33–16, the main program of the computer is interrupted for one machine cycle and the data word acquired from variable memory and placed into the C2 register for transfer to the output unit selected. At this time, a Channel 2 Ready signal is generated which, when applied to Ā33–19, sets FF33–21 indicating that the C2 register is now full. This signal indicates to the output unit involved that it may now sample the contents of C2 register. If the output unit is one that can accept a parallel 25 bit number from C2 register, then it subsequently generates a positive C2-Resume signal which, when applied via N33–25 and N33–33 to Ā33–32, again sets F33–10 to generate a request for the computer to furnish the second data word to be transferred under direction of the F2 register contents. On the other hand, if the output unit is either tape or typewriter, then the D register, subsequently to be discussed, is successively filled with 6 bit characters from the C2 register for ultimate transfer to the tape or typewriter which can only process 6 bits at a time. After the C2 register has been completely emptied by the D register operation, a signal DSC Output Resume is applied to Ō33–14 which allows Ā33–15 to request another data transfer interrupt of the main program in a manner similar to the request made by Ā33–11.

After the last data word has been withdrawn from memory and placed in C2 register, the Terminate FF33–34 is set which permits Ā33–35, upon request by the unit as indicated by a signal from Ā33–11, to generate a Battery Out Drop Go signal to set the Drop Go F2–13 in FIGURE 31. However, if a tape unit or typewriter is selected, then Ā33–35 cannot be enabled because of the negative signal F2 Group 7 And Go. In such case, F2–13 is set by a Tape or TW (Out) Drop Go.

When a data transfer input operation is being performed on channel 2, circuitry is provided in FIGURE 33 which differs somewhat from that provided for an output operation. For example, when FF33–10 is set, Ā33–38 is responsive to both it and the F2 Input And Go signal from FIGURE 31 in order to generate a signal Gate DTU to C2 which transfers a 25 bit data word from an input unit into C2 register. At the same time, Ā33–38 sets C Status FF31–21 indicating that the C2 register is filled so that the data word residing therein may be gated into O register and from there written into the variable memory. In order to accomplish the transfer of information from C2 to O register, the output from Ā33–38 is also applied to Ō33–16 to request an Input-Output Data Transfer operation of channel 2 so that an interrupt of one machine cycle is made of the main program to allow this single data word transfer to take place. When the data word is transferred from the C2 register to the O register, a Channel 2 Resume signal is generated which enables Ā33–41 to clear C Status FF33–21 thus denoting to the input unit that C2 register is available to receive another data word therefrom. A positive C2 Resume signal from the unit thereafter sets FF33–10 and the cycle repeats. However, after the required number of data words have been successively transferred between the input unit and the computer, the program circuitry sets the Terminate FF33–34 in order to allow Ā33–43, upon receipt of an output from Ā33–41, to set the Drop Go F2–13 and thus disable Ā33–32 due to the subsequent clearing of F2–14. Ā33–43 is disabled by the positive Tape and Go signal which is present when a tape input is being performed.

In the event that the input operation is utilizing a magnetic tape unit or the typewriter, the D sequence control unit associated with the D register provides a DSC Input Resume Signal directly to Ō33–16 to request an input data interrupt. In addition, the DSC Input Resume signal sets C status FF33–21 to be transferred into variable memory. The function of FIGURE 33 will be elaborated upon in detail in subsequent discussion.

FIGURE 34 shows details of the 7 bit D register (including parity) which is utilized as a buffer between the C2 register and the magnetic tape units and typewriter. Magnetic tape units associated with the computing system here disclosed are characterized by their bit handling capacity of 6 binary bits plus one parity bit. The electric typewriter unit is characterized by its bit handling capacity of 6 binary bits (not parity bit). During an output tape operation, the D register receives successive groups of 6 bit characters from the C2 register stages 0–5, 6–11, 12–17, and 18–23, in that order. An odd or even parity bit, as specified by the F2 register stage 02, is added to each 6 bit character residing in the D register by appropriately setting register stage D06. The content of D register is transferred to the selected tape via the L register subsequently to be described.

In FIGURE 34, a transfer into the D register stages 0–5 from C2 register stages 0–5 is accomplished by a set of Ā gates 34–22 through 34–27 upon occurrence of a gating signal Gate C2 (0) To D applied to N34–28. Likewise, 6 bits may be transferred to D0–5 from C2 register stages 6 through 11 respectively, via a set of Ā gates 34–29 through 34–34 upon occurrence of the gating signal Gate C2 (1) To D. A third group of 6 bits may be gated into D register from stages 12 through 17 of C2 register by Ā gates 34–36 through 34–41 upon generation of Gate C2 (2) To D. The last 6 bit group from C2 stages 18 through 23 is subsequently gated into D register via Ā gates 34–43 through 34–48 when the signal Gate C2 (3) To D is applied.

The D register may also be filled from either a tape unit or the typewriter in the following manner. Outputs from the read sense amplifiers respectively associated with the seven tape tracks are directly applied to the D register stages as shown. For example, track 7 of the magnetic tape, which contains the parity bit associated with each of the 6 bit characters thereon, is gated to D6. Tape tracks 1 through 6, in which are placed the six information bits associated with each tape character, are respectively applied to set appropriate ones of the D register stages 5–0 in the event that a binary 1 bit in the track is detected by the read amplifier. Ā gates 34–56 through 34–61 are similarly responsive to outputs from the typewriter which also generates a 6 bit binary code for each character to be entered into the D register. The signals from the typewriter are indicated by the SS–1 through SS–6 signals applied to the Ā gates 34–56 through 34–61 as shown in FIGURE 34.

When a data input operation is being performed utilizing either a tape unit or the typewriter, it is first necessary to completely fill the C2 register with a twenty-four bit number before gating the C2 register contents to the variable memory register O. This requires that the C2 register be successively filled from the D register with groups of 6 bits each, commencing with the C2 register stages 0–5. Thus, Ā gates 34–64 through 34–69 are responsive to the set output terminals of D0–5 in order to transfer the D register contents to the indicated C2 register stages upon occurrence of the signal gate D To C2 (0). This is the first 6 bit transfer made between the D register and C2 register, with D register being subsequently cleared in preparation for entry of a second 6 bit character from the selected input unit. After the second character is deposited in D register, Ā34–70 through 34–75 are enabled to a signal gate D To C2 (1) to transfer the D register contents to the indicated C2 register stages 6 through 11. Thereafter, the D register is cleared and the third 6 bit character from the input unit entered therein which is subsequently gated via Ā gates 34–76 through 34–81 into C2 register stages 12–17. A fourth character from the selected input unit is thereafter entered into D register and subsequently transferred via Ā gates 34–82 through 34–87 into C2 register stages 18–23. After the fourth group transfer, C2 register is now filled and may now be gated into O register for storage within the variable memory.

FIGURE 35 shows details of the D sequence control used for channel 2 input-output data transfers to or from tape units or the typewriter. A D Status FF35–10 is provided which, when set, indicates that the D register is filled with a seven bit character (including parity) which thereafter must be transferred either to the C2 register or to the output unit depending upon the direction of the data transfer. If a tape unit is selected for the output operation, a negaitve signal Not Initiate Tape Sequence is applied to Ō35–15 from the selected tape unit as soon as the tape is up to operating speed. On the other hand, the selection by the F2 register of a typewriter input-output data transfer causes the generation of a negative signal NOT TW AND Go from FIGURE 31 which is applied via N35–32 to various portions of the D sequence control in order to synchronize the typewriter with the computer. If a tape output operation is initiated, the output from Ō35–15 is generated as soon as the tape unit is up to speed and is applied to Ā35–16 so that the first character from C2 register can be gated into D register for transfer to the tape unit via L register. On the other hand, if a tape input operation is commenced, a tape character must first be placed into D register after which the D Status FF35–10 is set by the In Tape Resume signal from FIGURE 38. This causes Ā35–14 to gate D register into the appropriate group of stages of the C2 register. A detailed description of the D sequence control operation will be given in subsequent paragraphs.

FIGURE 36 shows details of the D counter which is a two stage, double level counter that counts the characters passing through the D register during a tape or typewriter input or output operation. It generates the groups of signals gate C2 To D and gate D To C2 which are respectively used in the D register of FIGURE 34 in order to transfer a 6 bit character between C2 register and the D register. The condition of the two lower flip-flops FF36–10 and 36–11 determines the condition of the two upper flip-flops FF36–12 and 36–13. This is accomplished at MP4 time such that Ā36–15 and Ā36–16 respectively set FF36–12 and FF36–13 if their associated flip-flops FF36–10 and FF36–11 are set at this time. Conversely, Ā36–17 and Ā36–18 respectively clear FF36–12 and FF36–13 if at MP4 time their associated flip-flops FF36–10 and 36–11 are cleared. The conditions of FF36–12 and FF36–13 are sensed by several output Ā gates which generate the required gating signals. For example, Ā36–21 senses the cleared condition of both FF36–12 and 36–13 to generate a signal Gate C2 (0) To D at the time that a signal Gate C2 To D is applied via N36–25. In similar fashion, Ā36–22 is responsive to the set condition of FF36–12 and the clear condition of FF36–13 to generate the signal Gate C2 (1) To D if N36–25 generates a significant positive output therefrom. Ā36–23 and Ā36–24 respectively generate the signals Gate C2 (2) To D and Gate C2 (3) To D if the contents of FF36–12 and 36–13 are respectively 01 or 11. If the signal Gate D to C2 is applied to N36–30 then Ā gates 36–26 through 36–29 respectively generate one of the four signals Gate D to C2 as shown which are used to transfer successive 6 bit characters from the D register into one of the four groups of C2 register stages.

In order to change the condition of FF36–10 and 36–11, which thereafter change the condition of FF36–12 and 36–13, Ā gates 36–31, 36–32, 36–34, and 36–35 are utilized in the manner subsequently to be described. The output of Ā36–32 is also applied via N36–33 to the inputs of Ā36–34 and Ā36–35. Ā36–31 and Ā36–34 also have respective inputs from the clear output terminals of their respectively associated upper flip-flops 36–12 and 36–13, while Ā36–32 and Ā36–35 have inputs from the set terminals of their respectively associated flip-flops 36–12 and 36–13. Ā36–31 and Ā36–32 also have inputs to which a positive Advance DK-Lower signal is applied from FIGURE 35 at the time that a change in the states of FF36–10 and FF36–11 is desired. This advance signal can only be applied at MP12 time of a cycle such that the condition of the two lower DK flip-flops change prior to application of MP4 to Ā gates 36–15 through 36–18. Ā36–19 is responsive to the clear condition of both FF36–10 and FF36–11 to generate an output signal via N36–20 which, when positive, indicates that the value in the lower flip-flops of the D counter is equal to 0.

Initially, a Channel 2 Clear signal is applied from FIGURE 35 prior to initiation of an Input-Output operation thereon, which clears both FF36–10 and FF35–11 such that the DK=0 signal is now positive. Subsequent to the clearing of the lower flip-flops of the D counter, a MP4 pulse applied to Ā36–17 and 36–18 also clears FF36–12 and 36–13 such that Ā36–21 or Ā36–26 is conditioned to pass the respective signal Gate C2 To D or Gate D To C2 depending on the direction of transfer. If now a positive Advance DK-Lower signal is applied at MP12 time of a machine cycle, Ā gate 36–31 is responsive both to it and to the positive output from the clear terminal of FF36–12 to set FF36–10 only. Thereupon, the next subsequent MPA pulse applied to the transfer gates will set FF36–12 but maintain FF36–13 in its clear condition inasmuch as FF36–11 is clear. Under such a situation, the set condition of FF36–12 and the clear condition of FF36–13 condition Ā gates 36–22 and 36–27 to pass the generated one of their respective gating signals. When the next following Advance DK-Lower signal is applied to Ā gates 36–31 and 36–32, only Ā36–32 is enabled since it has applied thereto the positive output from the set terminal of FF36–12. Therefore, the output from Ā36–32 clears FF36–10 and also is applied via N36–33 to the input of Ā36–34. With respect to the latter, it is further responsive to the positive signal from the clear output terminal of FF36–13 so that it sets FF36–11. The next following MP4 pulse gates the cleared condition of FF36–10 to FF36–12 and the set condition of FF36–11 to FF36–13 such that Ā gates 36–23 and 36–28 are now conditioned. The third positive advance pulse applied to the D counter consequently finds Ā36–31 enabled to set FF36–10, so that at the next following MP4 time, the set conditions of both FF36–10 and FF36–11 are gated via Ā36–15 and Ā36–16 to set the respective flip-flops 36–12 and 36–13. The setting of these thereby causes the generation of either a Gate C2 (3) To D or a

Gate D To C2 (3) at the appropriate time, depending on whether the transfer is either an output or input.

The fourth following advance pulse to be received by the D counter finds $\overline{A}$**36–32** enabled to clear **FF36–10**, with this output also being applied to $\overline{A}$**36–35** to clear **FF36–11**. The next subsequent MP4 gates the cleared condition of both of the lower D counter flip-flops to the upper flip-flops whereupon the cycle is completed and $\overline{A}$ gates **36–21** and **36–26** are again enabled. Thus, the D counter is seen to be generally a binary counter of two stages having four unique combinations of the two binary bits.

FIGURE 37 shows details of the L register whose primary function is to convert the data contained in the D register into a Non-Return-To-Zero, Change On 1 tape output (write) operation. During an output operation, the set output terminals of each of the upper row of stages **L00** through **L06** is connected directly to one of the write amplifiers in the selected tape unit. Each stage of this register fixes the direction of magnetization on its associated track of the tape. In the Non-Return-To-Zero, change on 1 method of digital recording, no fixed state of magnetization is assigned to either the binary 1 or 0 bit value held in an L register stage. Rather, the state of magnetization is reversed each time the digit 1 is to be recorded on the tape track. In sensing a character from a tape during the read operation, each change of flux thus indicates that a binary 1 is recorded in the bit position.

In FIGURE 37, the set output terminals of each of the D register stages **0** through **6** are respectively applied to $\overline{A}$ gates associated with the correspondingly numbered L register stages. For example, **D0–6** are connected to a group of $\overline{A}$ gates **37–10** through **37–16** whose outputs are respectively connected to the said input terminals of **L0–6**. D register stages **0–6** are also connected to a second group of $\overline{A}$ gates **37–17** through **37–23** which in turn are used to clear the respective L register stages **0** through **6**. The contents of the upper row of L stages **0–6** may also be gated to a lower row of L* stages **0** through **6** via a set of $\overline{A}$ gates **37–25** through **37–31** at MP12 time of a cycle. For example, if **L00** is set at the time that MP12 occurs, then L***00** will likewise be set at this time. The set output terminals of L***0–6** will likewise be set at this time. The set output terminals of L***0–6** are connected to respective ones of the $\overline{A}$ gates **37–17** through **37–23** which are used to clear **L0–6**. The clear output terminals L*0–6 are applied to the first group of $\overline{A}$ gates **37–10** through **37–16**. L***0–6** is unconditionally cleared at each MP10 time just prior to entry of information from **L0–6** at MP12. Both **L0–6** and L***0–6** are cleared upon application of either a clear L or a Channel Z Clear signal applied to $\overline{O}$**37–32**.

The manner in which the L register acts to convert the contents of D register into Non-Return-To-Zero, Change On 1 indications will now be described. Assume that the **C2** register holds a 24 bit number which is to be transmitted via the D register to a selected tape unit for recording 6 bits at a time. If a Channel **2** Clear signal is applied to $\overline{O}$**37–32**, both **L0–6** and L***0–6** are cleared. Therefore, the negative signal coming from the set output terminals of **L0–6** causes the tape write amplifiers to magnetize each tape track in the same direction. Assume that the first 6 bits from **D0–5** have a configuration 100110 (the left most bit being held by **D0**). D register is gated to the input gates of the **L0–5** stages, together with a generated parity from D register stage **6** to **L06**. The parity bit is ignored in the following discussion. Since L***0–5** is at this time cleared, $\overline{A}$ gates **37–10** through **37–15** are enabled to pass exactly the D register stage conditions such that **L0–5** now contains the number 100110. The change in the condition of L stages **0**, **3** and **4** are shown in Table 6 below, will thus cause the associated tape write amplifiers to reverse the magnetization of their associated tape tracks, with this reversal indicating that binary 1's are in this character position.

TABLE 6

	D0-5 Stages	L*0-5 Stages	L0-5 Stages
	000000	000000	000000
1	100110	000000	100110
2	011100	100110	111010
3	100100	111010	011110
4	110011	011110	101101
5	000000	000000	000000

Subsequent to the operation above described, the contents of **L0–5** are gated to L***0–5** via $\overline{A}$ gates **37–25** through **37–31** such that the latter also holds a configuration 100110 as indicated in row 2 of Table 6. If the D register is now cleared and the second 6 bits from **C2** register placed therein (having, for example, a configuration of 011100 as indicated in row 2 of Table 6) the subsequent gating of the D register contents to L register sets the L register stages **0–5** to a configuration shown in row 2 of Table 6. This is so, since none of the signals from D register stages **0**, **4**, or **5** are positive inasmuch as the stages contain binary 0's. Thus, neither of the $\overline{A}$ gates associated with the set and clear input terminals of the corresponding L register stages **0**, **4**, and **5** will be enabled so that these L register stages remain undisturbed. However, since in row 2 it is seen that **D1** holds a binary 1, the positive signal therefrom is applied, together with the positive signal from the clear output terminal of L***01**, enable $\overline{A}$**37–11** to set **L01**, thus reversing the magnetization of the associated tape track **5**. A similar operation occurs with respect to the setting of **L2** by the 1 bit contained in **D2**. Since **D3** also contains a binary 1, the positive signal therefrom switches with the positive signal from the set output terminal of L***03** to enable $\overline{A}$**37–20** to clear **L03** from its previous set condition. Therefore, tape track **3** also reverses its direction of magnetization, thereby indicating the writing of a binary 1.

Subsequent to the setting of **L0–5** to the configuration 111010, its contents are transferred back to L* register in preparation for entry of the third group of 6 bits from the D register, which entry is assumed to be 100100 as indicated in row 3. Therefore, since binary 1's are only found in **D0** and **D3**, only **L0** and **L3** conditions will be reversed, so that the L register configuration is now 011110 as shown in row 3 of Table 6. After transfer of this L register configuration of L*, the fourth 6 bit group from D register is applied to the L register (assumed to be 110011 as shown in row 4). This setting of the D register, coupled with the setting of L* to a configuration of 011110, reverses the conditions of **L0**, **L1**, **L4**, and **L5** such that the configuration of **L0–5** is now 101101.

In the event that the input-output data operation calls for the successive recording of two or more 24 bit data words from variable memory, the **C2** register content is replaced with the second data word and that D register continues to supply 6 bit characters to the L register. After all of the data words have been recorded on the tape a Clear L signal is applied to $\overline{O}$**37–32** to clear all of the **L0–5** register stages. The reversal of magnetization due to this final clearing at the end of a tape record is thereby interpreted as a check mark which is used during a subsequent read operation of the same record to enable performance of a longitudinal parity check on drop outs. For example, assume in Table 6 that a tape record comprises only one 24 bit data word from variable memory, which is divided into four 7 bit characters (including parity) successively recorded upon the tape. A clear signal is applied to L register stages **0–5** (ignoring parity in this example) to effect a reversal of magnetization in certain of the tape tracks. When this same record of four characters is subsequently read for entry back into the variable memory of the computer, the L register has a secondary function to perform a longitudinal parity check on drop outs. Thus, during a tape read operation, each 6 bit character placed in D register is also gated therefrom to L register. However, the output from the L0–6 stages at this time is not allowed to write on the tape. The transfer from **L0–6** to L*0–6 is also performed during the read operation as in the write operation. When the D register is set according to the 1 bits comprising the check mark, which in the case of Table 6 would have a configuration of 101101, this check mark is thereafter gated to the L register. If there have been no longitudinal bit drop outs during the read operation, the L register stages **0–6** now should be all in their clear condition. If this is not the case, then an error is generated. $\bar{O}$ gate 37–37 is responsive to the clear output terminals of all of the L stages **0–6** so that the presence of a binary 1 bit in any one of them will result in a significant positive output being generated therefrom.

FIGURE 38 shows the line pulse generator circuits used primarily for spacing data on a tape during a tape output operation. The magnetic tape units associated with the present system are characterized by their word transfer rate of 140 microseconds per word, with such a word being comprised of four 7 bit characters (including parity) spaced successively along the length of a tape. Therefore, recording and read time between successive characters is approximately 35 microseconds, or seven of the computer machine cycles. When characters are read from a tape, there is a 35 microsecond interval between the sensing of any reversal of magnetization on the tape indicating the presence of binary **1**'s.

In FIGURE 38, flip-flops **38–11** through **38–16** comprise a counter for timing a seven machine cycle interval between characters in a tape output operation. **FF38–11** through **38–13** comprise the lower row in the counter while FF38–**14** through **38–16** comprise the upper row of the counter. The content of each lower row flip-flop is gated to the corresponding upper row flip-flop by one of a set of $\bar{A}$ gates **38–18** through **38–23**, with the first three $\bar{A}$ gates of this group sensing the clear output terminal of their associated lower row flip-flop while the last three $\bar{A}$ gates of this group sense the set output terminals of the lower row of flip-flops. The transfer between lower and upper rows occurs at MP2 time of each machine cycle. The condition of the lower row flip-flops **38–11** through **38–13**, and thus the condition of the upper row flip-flops, can be changed in binary progression form during each successive machine cycle by $\bar{A}$ gates **38–24** through **38–30**, and N38–**32**. As shown, $\bar{A}$ gates **38–24** through **38–26** are respectively connected to the set input terminals of FF38–**11** through **38–13**, while $\bar{A}$ gates **38–27** through **38–29** are respectively connected to their clear input terminals. A counter advance signal is applied via **N38–31** to $\bar{A}$ gates **38–24**, **38–27**, and **38–28** to accomplish the actual counting operation at MP0 time.

Table 7 below indicates the states of the counter flip-flops in FIGURE 38 for eight successive machine cycles, during which time a counter advance pulse is applied at each MP0 time and the contents of **FF38–11** through **FF38–12** are shifted upwards at **MP2**.

TABLE 7

MC	FF38–11	FF38–14	FF38–12	FF38–15	FF38–13	FF38–16
0	Clear	Clear	Clear	Clear	Clear	Clear.
1	Set	Set	Clear	Clear	Clear	Clear.
2	Clear	Clear	Set	Set	Clear	Clear.
3	Set	Set	Set	Set	Clear	Clear.
4	Clear	Clear	Clear	Clear	Set	Set.
5	Set	Set	Clear	Clear	Set	Set.
6	Clear	Clear	Set	Set	Set	Set.
7	Set	Set	Set	Ser	Set	Set.
8	Set	Set	Clear	Clear	Clear	Clear.

Assume first that all six counter flip-flops are cleared, as shown in row 0 of Table 7. If $\bar{A}$**38–36** now generates a signifiicant output at MP0 time, this is inverted via **N38–31** and applied to the $\bar{A}$ gates shown in FIGURE 38. The first such advance signal finds $\bar{A}$**38–24** conditioned by a positive signal from the clear output terminal of **FF38–24** conditioned by a positive signal from the clear output terminal of FF**38–14** so that said $\bar{A}$ gate is enabled to set FF**38–11**. At MP2 time, the content of FF**38–11** through **38–13** is gated to the upper row flip-flops such that **FF38–14** is now set while **FF38–15** and **FF38–16** remain clear. This condition is indicated in row **1** of Table 7. The second advance pulse at the next following **MP0** time causes $\bar{A}$3**8–27** to be enabled to clear FF**38–11**, due to the fact that N3**8–37** generates a positive signal from the set condition of FF**38–14**. At this time also, $\bar{A}$**38–30** produces a positive output since at least one of the flip-flops **38–15** and **38–16** is cleared. $\bar{A}$**38–25** is also enabled by this advance pulse to set FF**38–12** since at this time F**F38–15** is in its cleared condition and so produces a positive signal at its clear output terminal. Therefore, by MP2 time of the second machine cycle under consideration, FF**38–15** is set while **FF38–14** and **FF38–16** are both clear.

For each of the succeeding machine cycles the advance pulse causes a change in the conditions of the lower and upper row flip-flops in accordance with rows 3 through 7 of Table 7. It is believed that a detailed explanation of this change is not necessary in view of the description given above. The seventh advance pulse applied to the counting circuit results in all of the upper row of flip-flops being set by MP2 time, as indicated in row 7. The next following advance pulse, however, maintains **FF3–14** in its set condition while clearing **FF38–15** and **FF38–16.** This is due to the fact that the set conditions of both FF3**8–15** and **38–16** cause $\bar{A}$38–30 to generate a negative signal which disables $\bar{A}$38–27, so that the latter cannot clear FF3**8–11** upon receipt of the next following advance pulse. This advance pulse does enable both $\bar{A}$**38–28** and $\bar{A}$**38–29** to generate signals clearing FF**38–12** and **38–13**, respectively. Therefore, row 8 is the same as row 1 of Table 7. If further advance pulses are received, the count cycle repeats itself according to rows 2 through 7, and back to 1.

In order to enable $\bar{A}$**38–36** to generate the advance pulses for the line pulse generator counter, the Line Pulse Generator FF38–10 must be set. When cleared, this flip-flop enables $\bar{A}$**38–17** to generate clear signals for flip-flops **38–11** through **37–13**. FF38–**10** may be set in a variety of ways both during tape input and output operations. For example, when a tape output operation is being performed, the selected tape unit generates a negative signal Start LPG as soon as it is running at its designed operating speed. During a tape input operation, i.e. tape characters are successively read from tape and stored into the D register, Line Pulse Generator **FF38–10** is set as soon as a tape character enters the D register. $\bar{O}$**39–90** detects the entry of a character in D register and generates a positive signal First Bit which enables $\bar{A}$**38–64** to set the Tape Resync **FF39–63**. Thereafter at MP14 time, $\bar{A}$**38–71** is enabled to set **FF38–10**. The first count pulse generated from $\bar{A}$**38–36** is also used to clear FF38–**63** which will be set again by entry of the next following tape character into the D register. However, as long as FF**38–10** is set, $\bar{A}$**38–64** is inhibited from setting FF38–**63**, so that the same tape character in **D** register cannot set the Tape Resync **FF38–63** more than once.

$\bar{A}$**38–35** is responsive to the set condition of **FF38–14** and the clear conditions of **FF38–15** and **38–16** (as shown in row 1 of Table 10) to generate a signal applied to $\bar{A}$ gates **38–39**, **38–40**, and **38–41**. $\bar{A}$ gate **38–35** must also be conditioned by the **F2** Output And Go signal from

FIGURE 31 which indicates that an output data transfer operation is being performed. **Ā38–39** generates a signal at MP4 time which inserts a parity bit into **D6**. At MP6 time of the same machine cycle, **Ā38–40** generates a signal via **Ō38–43** which gates the contents of D register to the L register so that the latter enables the tape write amplifiers to record the character on the tape tracks. At MP8 time of this cycle, **Ā38–41** generates a signal Out Tape Resume which first clears the D register of the character just recorded and then initiates a transfer into D register of the second six bit character from C2 register.

Ā38–34 is responsive to the set condition of both **FF38–14** and **FF38–15**, and to the clear condition of **FF38–16** to generate a second enabling signal which is applied via **N38–45** to Ā gates **38–46**, **38–47**, **38–48**, **38–52**, and **38–53**. The enabling of **Ā38–34** thus occurs during the third count cycle indicated by row 3 of Table 7. In the event that an input data transfer is being performed, then **Ā38–46** at MP8 time transfers the contents of D register into L register to allow the latter to check for longitudinal drop outs as previously described. At MP10 time, **Ā38–47** generates the In Tape Resume signal which is used to transfer the 6 bit character contained in D register to the C2 register. At MP15 time, **Ā38–52** is enabled, if the End of Record **FF38–62** has been set, to test the contents of **L0–6** in order to generate an L Parity Error signal from **Ā38–55** in the event that not all of the L register stages are in a clear condition. The determination of the end of record during a tape read operation will be subsequently discussed.

When all of the flip flops **38–14** through **38–16** are in their set conditions, such as indicated in row 7 of Table 7, **Ā38–33** is enabled if an output operation is being performed (indicated by the presence of the F2-Output And Go signal from FIGURE 31). The enabling of **Ā38–33** permits **Ā38–50** to sample D register to determine whether it contains another 6 bit character to be transferred to the tape write amplifiers via the L register. If the D register is empty at this time, all computer words comprising the record have at this time been withdrawn from variable memory and written on the tape. Therefore, the tape write operation now ceases until there is required another output channel **2** operation by a different instruction Initiate F2. The output from **Ā38–50** is applied via **Ō38–49** to clear Line Pulse Generator **FF38–10** which thus disables **Ā38–36** from generating counter advance pulses. In addition, the output from **Ā38–50** is also applied via a 75 microsecond delay **38–60** to **Ā38–61** which subsequently sets **FF38–10** as well as End Of Record **FF38–62**. The setting of **FF38–10** again enables **Ā38–36** to advance the counter through one last complete sequence. **FF38–62**, when set, thereby permits **Ā38–51** at MP15 time of the last count cycle (when **FF38–14**, **38–15**, and **38–16** are all set) to again clear **FF38–10** via **Ō38–49** as well as to generate a Clear L signal which, when applied to the L register, causes the check mark to be placed on the tape at the end of the record. Therefore, the check mark is placed approximately 135 microseconds behind the last data character in the record.

During a tape input operation, the F2-Input And Go signal is positive instead of the F2-Output And Go signal previously considered. Therefore, Ā gates **38–35** and **38–33** are inhibited from providing significant negative outputs during the counter sequencing operation. However, **Ā33–34** responds only to the condition of flip-flops **38–14** through **38–16** and so will be enabled when these three flip-flops are respectively in the conditions 1, 1 and 0 (see row 3 of Table 7). For this cycle of the counter, Ā gates **38–46** and **38–47** are respectively enabled at MP8 and MP10 times as shown in FIGURE 38. The output from **Ā38–46** is applied to **Ō38–43** to gate the tape character in D register into L register so that the latter may perform its secondary function of longitudinal parity check. Furthermore, the signal **I** Tape Resume from **Ā38–47** is applied to set D **FF35–10** indicating that D register now contains a 6 bit tape character (excluding parity) which must be transferred to the appropriate group of the C2 register stages in accordance with the value held in D counter. During a tape input operation, the signal from **Ā38–34** also enables **Ā38–52** to clear Line Pulse Generator **FF38–10** via **Ō38–49** and **N38–60**. When **FF38–10** is cleared, **Ā38–17** clears all of the counter stages **38–11** through **38–13**. Therefore, during a tape input operation the line pulse generator counter is not allowed to run through a complete sequence as is the case for a tape output operation. **Ā38–52** also is applied via **N38–54** to **Ā38–55** which in turn determines if L register contains a value other than 0. However, **Ā38–55** is only enabled when End Of Record **FF38–62** is set. If such is the case, then the determination that any one of the L register stages is set indicates that there has been a longitudinal parity error during the read operation of the tape record. **Ā38–56** is also enabled at the end of record to stop the selected tape unit by generation of the various signals Clear Transmit Tape Sequence, Set Read Stop Delay, etc.

The function of the end of record detecting circuits for a tape read operation will now be described. End Of Record **FF38–62** is set subsequent to the reading of the last data character in a tape record and prior to the sensing of the check mark at the end of the record. As mentioned before, during a tape read operation, Line Pulse Generator **FF33–10** is set by each tape character as it enters the register, and is subsequently cleared approximately 15 microseconds later by an output from **Ā38–52**. Since tape data characters within a record are spaced approximately 35 microseconds apart at the normal tape operating speed, it is therefore seen that **FF38–10** is always cleared by the time that the next subsequent tape character is read. **FF38–10** in its cleared condition thereby conditions **Ā38–64** to respond to the entry of the next tape character as previously described. **FF38–10** when set also enables **Ā38–68** to produce a negative output which is applied to the resettable delay circuit **38–69**. Delay **38–69** is of the type which is responsive to a negative signal leading edge to generate an output, after 75 microseconds, which is sufficient to set End of Record **FF38–62**. However, if a second and succeeding negative signal leading edge is applied to the delay circuit prior to the expiration of this 75 microsecond period, then the output signal therefrom will be delayed for an additional 75 microseconds. This type delay circuit is sometimes called a holdover circuit, and is well known in the art. As long as data characters from a tape record are being sensed and transmitted to the D register, the Line Pulse Generator **FF38–10** will always be set approximately every 35 microseconds since this is the time interval between successive data characters within a record. Each time that **FF38–10** is set, **Ā38–68** is enabled to generate a negative signal therefrom which thereupon resets delay **38–69** and prevents a set pulse from being applied to **FF38–62**. However, there is approximately a 135 microsecond interval between the last data character of a record and the check mark associated with that record as has previously been described in connection with the tape write operation. Therefore, following the last data character to be read from the tape record, no negative signal will be generated from **Ā38–68** for at least 75 microseconds. This enables the setting of **FF38–62**. **FF38–62**, when set, applies a positive signal via **N38–65** to a 100 microsecond delay **38–66** to enable **Ā38–67** to again set **FF38–10** in preparation for reading the check mark at the end of the record. **Ā38–67** must be included in the circuit to insure the setting of **FF38–10** at this time, since an End Of Record check mark does not necessarily have to have any binary 1's therein as does each of the data characters. For example, if a check mark consists of binary 0's, then none of the D register stages would be set so there would be no indication of a first bit from **Ō34–90.** Upon **FF38–10** being set for the last time, advance pulses are applied to the counter flip-flops so that **Ā38–52** is eventually enabled by an output from **Ā38–34.** The negative signal from **Ā38–52** is applied via **Ō38–49** to again clear **FF38–10.** In addition, it is applied to **Ā38–55** which is now enabled by the set condition of **FF38–62** to sample L register for binary 1's in any of its stages. Since the check mark had previously been gated from D register to L register by the signal from **Ā38–46,** the longitudinal parity check is now completed and an L Parity Error indicated if L register does not have binary 0's in all of its stages.

A description will now be given of the cooperation between the units shown in FIGURES 34 through 38 for a Channel **2** output transfer to one of the tape units. When such an operation is being performed under control of a number held in F2, a positive signal F2 Output And Go is continuously generated from FIGURE 31 which is applied to, among others, Ā gates **35–16** and **35–17** in the D sequence control of FIGURE 35. Furthermore, a negative signal is applied to **Ō35–15** indicating that a tape unit has been selected and is waiting for information from the computer. D Status **FF35–10** is assumed to be cleared prior to the transfer of the first word from O register to C2 register. All flip-flops in the D counter are also cleared at this time in preparation for the subsequent gating of C2 register stages **0–5** into the D register. Upon the C2 register being filled with information from the O register, a positive C2 Full signal is applied to the D sequence control unit. This signal thereby enables **Ā35–16** to set Trigger D Status **FF35–11** at MP12 of a machine cycle. The output from **Ā35–16** also generates the signal Gate C2 To D which, when applied to **Ā36–21** causes C2 register stages **0–5** to be transferred into the D register. **Ā35–16** also enables **Ō35–24** to set the D counter flip-flop **36–10** in a manner previously described. However, the cleared status of **FF36–12** and **36–13** remains undisturbed at this MP12 time to insure that **Ā36–21** passes the Gate C2 To D signal. Since **FF36–10** is set, **Ā36–19** via **N36–20** no longer generates the positive signal DK=0. Therefore, the enabling of **Ā35–17** at MP14 time cannot cause **Ā35–19** to generate the signal DSC Output Resume. This latter signal appears only after the complete word in C2 register has been transferred to the tape unit. However, the output from **Ā35–17** does set the D Status **FF35–10** which thereby terminates the positive D Empty signal from **N35–23.** The positive output from the set terminal of **FF35–10** also enables **Ā35–20** to direct a signal via **Ō35–21** to **Ā35–22** to clear **FF35–11** at the next following MP0 time. However, as long as the D Status **FF35–10** is set, **Ā35–16** is unable to again set Trigger D status **FF35–11.**

The D register now contains the first character to be transferred to the selected tape unit, said character being a portion of the first 24 bit word involved in the data output transfer. In FIGURE 38, Line Pulse Generator **FF38–10,** which is initially cleared by a Channel **2** Clear pulse at commencement of the output operation, remains so until a negative Start LPG signal is transmitted from the tape unit indicating that it is ready to receive a character. This negative signal sets **FF38–10** which thereafter enables **Ā38–36** to set **FF38–11** at the next following MP0 time. At MP2, **Ā38–35** is enabled by the corresponding setting of **FF38–14.** Thus, Ā gates **38–39** through **38–41** now generate signals at MP4, MP6, and MP8 times, respectively. The signal from **Ā38–39** inserts a parity bit for the character held in D register, said character being subsequently gated from D register to L register by **Ā38–40.** When in the L register, the character is recorded on the tape in the manner previously described, with "1" bits causing a reversal of the magnetization. At MP8 of this machine cycle, **Ā38–41** generates the Out Tape Resume signal which clears D register via **Ō35–30** and the D Status **FF35–10.** The clearing of **FF35–10** now permits **Ā35–16** to be enabled at the next following MP12 time to again set **FF35–11** and gate **C2** register stages **6–11** into the now empty **D** register. At this MP12 time, **FF36–12** is in a set condition while **FF36–13** is in a cleared condition so that a signal gate **C2** (1) To D is generated from **Ā36–22.** This signal thereupon transmits the binary information in register stages **6** through **11** of the C2 register into the D register where they comprise the second character to be transmitted to the tape unit. In addition, the signal from **Ā35–16** is applied via **Ō35–24** to advance the lower flip-flops in D counter. At MP14 time, Ā gate **35–17** is again enabled to set the D Status **FF35–10** which thereafter remains in this condition until D register can be emptied of the second tape character.

The line pulse generator in FIGURE 38 continues to step through its seven cycles in a manner previously described after which it once again returns to cycle **1.** By the time that **Ā38–33** is enabled, it will be noted that D register has been filled with the second tape character as above described so that the D Empty signal from FIGURE 35 is negative, thus preventing **Ā38–50** from generating a significant negative output. Consequently, there is no output from **Ō38–49** which in turn allows **FF38–10** to be maintained in its set condition. Advance pulses from **Ā38–36** are thus enabled each MP0 time to step the line pulse counter circuits through their maximum sequence and back to the cycle **1** as above described. **Ā38–35** will thereupon be enabled again in order to allow Ā gates **38–39** through **38–41** to again insert a parity bit for the second tape character which in turn is then gated to the L register for recording on the tape. The Out Tape Resume signal from **Ā38–41** will again cause D Status **FF35–10** to be cleared so that the third tape character can subsequently be placed into D register from C2 register stages **12** through **17.** The third tape character remains in the D register until the counting stages of the line pulse generator again recycle to a condition which enables **Ā38–35.** When such occurs, the third tape character from D register is again gated to L register, with the signal Out Tape Resume from **Ā38–41** subsequently allowing C2 register stages **18–23** to be gated into D register as the fourth tape character.

The gating of the fourth tape character into D register completes the transfer of the word held in C2 register so that the latter may now be cleared and the second word from variable memory transferred thereto if such is required by the input-output instruction. When **Ā35–16** is enabled at MP12 time to transfer the fourth tape character into D register, the Advance DK Lower pulse from **Ō35–24** results in both flip-flops **36–10** and **36–11** being returned to their cleared condition, since the maximum count of the D counter is 4. Since the output from **Ā35–16** also sets Trigger **D** Status **FF35–11** at this time, **Ā35–17** is again enabled at MP14 time. Because now the signal DK=0 is positive, **Ā35–19** is enabled to generate a significant signal DSC Output Resume which is applied to the **C2** sequence control of **FIGURE 33** to inform it that a new word may now be transferred into C2 register from the O register. If subsequent words are to be transferred, the Terminate **FF33–34** is at this time in its clear condition so that the significant negative signal DSC Output Resume enables **Ā33–15** to generate a Request Input Output Data Transfer in Channel **2.** This signal from **Ō33–16** is directed to the priority circuits which thus initiate a one machine cycle interruption of the main program in order to extract the second data word from the variable memory and place same in C2 register in the manner before described. Since the computer is immediately able to grant the request for this interrupt operation, then C2 register is filled with the second data word prior to the time that **Ā38–40** gates the fourth tape character from D register into L register. Thus, a positive C2 Full signal is maintained to **Ā35–16** at the MP12 time following the clearing of **FF35–10** by **Ā38–41.** Therefore, a signal from **Ā35–16** again causes the information of C2 register stages **0** through **5** to be gated into D register as the fifth tape character, with said fifth tape character comprising a portion of the second data word to be transferred in the output operation.

When no more data words are to be transferred into C2 register from the variable memory, then the signal C2 Full will be negative by the time that the last character is transferred from D register to L register. In this case, Trigger D Status **FF35–11**, and consequently D status **FF35–10**, cannot be set in the manner previously described. Therefore, when the counting stages in the line pulse generator of FIGURE 38 eventually attain a condition such that **Ā38–33** is enabled, **Ā38–50** senses the fact that D register is empty. A signal is generated from **Ā38–50** which is transmitted via **Ō38–49** to clear Line Pulse Generator **FF38–10.** The clearing of **FF38–10** thereupon causes **Ā38–17** to clear all of the stages in the line pulse generator counter. The output from **Ā38–50** also is supplied via the 75 microsecond delay **38–60** to subsequently set again **FF38–10** as well as setting the End of Record **FF38–62. Ā38–36** is now enabled once more to apply advance pulses to the line pulse counting stages so that upon **Ā38–33** being enabled, the positive signal from the set output of **FF38–62** enables **Ā38–51** to generate a signal at MP15 time which clears the L register and thus records the check mark on the tape. This output from **Ā38–51** is also applied via **Ō38–49** and **N38–60** to again clear **FF38–10.**

A brief description will now be given of the operation of D register, D counter, D sequence control, and Line Pulse Generator when a select tape unit transmits information to the variable memory of the computer. Assuming that all control flip-flops in FIGURE 38 are cleared prior to commencement of this operation, the first bit of the first tape character to arrive in D register from the tape read amplifiers causes **Ō39–90** to generate a signal enabling **Ā38–64** to set Tape Resync **FF38–63** at MP12 time. Thereafter, **Ā38–71** is enabled at MP14 time to set Line Pulse generator **FF38–10** to initiate operation of the line pulse generator counter flip-flops. Since a positive F2 Output And Go signal from FIGURE 31 is no longer applied to **Ā38–35, Ā** gates **38–39** through **38–41** are not enabled during the input operation. However, **Ā38–34** has no such restriction on it so the **Ā38–46, Ā38–47,** and **Ā38–52** are enabled during row 3 of Table 7. **Ā38–41** gates the first tape character from D register into L register so that the latter can perform its secondary function of checking for longitudinal drop outs as previously described. **Ā38–47** at MP10 time generates the signal In Tape eRsume which sets D Status **FF35–10** indicating that the D register is filled with a character waiting to be transferred to the C2 register. The setting of **FF35–10** thereupon enables **Ā35–14** to set Trigger D Status **FF35–11** to check the lateral parity of the tape character via **N35–29,** to gate the contents of D register into C2 register stages **0–5,** and to advance the lower flip-flops of thes D counter to a count of **1** in preparation for transfer of the second tape character, subsequently to be received, into the second group of C2 register stages.

At MP15 time following the generation of the In Tape Resume signal from **Ā38–47, Ā38–52** is enabled to clear **FF38–10** via **Ō38–49** and **N38–60.** The clearing of **FF38–10** thereby enables **Ā38–17** to clear the counting flip-flops **38–11** through **38–13** so that the line pulse generator counter cannot be stepped to its full count of 7. The previous setting of **FF38–10** also enabled **Ā38–68** to commence charging of delay **38–69** which in turn will set **FF38–62** at the end of 75 microseconds unless **FF38–10** is again set before that time by another tape character entering D register. Since a record on a tape consists of at least four characters, or a multiple thereof, the second tape character will enter D register prior to the completion of this 75 microsecond delay so that the first bit signal from FIGURE 34 sets **FF38–63** which in turn sets again **FF38–10.** The setting of **FF38–10** by the second tape character thereupon resets delay **38–69** so that the setting of **FF38–62** is postponed for at least another 75 microseconds.

When FF38–10 is set, pulses from **Ā38–36** again begin to advance flip-flops **38–11** through **38–13** to thereafter enable **Ā38–46** to gate the second tape character from D register to L register for performance of the longitudinal parity check, as well as generating the In Tape Resume signal from **Ā38–47** which in turn sets D Status **FF35–10.** After **FF35–10** is set, D register is gated to C2 register stages **6–11** and the D counter is advanced.

The above operation is repeated until the fourth tape character is gated from D register into C2 stages **18–23** by the enabling of **Ā35–14** at MP12 time of a cycle. At the next following MP14, **Ā35–26** is enabled to generate a signal DSC Input Resume from **Ā35–27.** This signal is applied via **Ō33–16** to the computer and requests it to interrupt the main program for one machine cycle to transfer the contents of C2 into the variable memory via the O transfer register. DSC Input Resume also sets **FF33–21** indicating that the C2 register is filled with information.

If the tape record being scanned contains a second group of four characters (a second computer word) the line pulse generator operates as described above to sequentially fill the C2 register with said second group of four characters. After this second group is placed in C2 register, **Ā35–26** again attempts to generate the DSC Input Resume from **Ā35–27.** However, if the computer requires only one word to be transferred thereto for this particular input operation, the Terminate **FF33–34** is set immediately subsequent to the storing of the first word from C2 register. Therefore, by the time that C2 register is loaded with the second word, the negative signal from the clear output terminal of **FF33–34** prevents **Ā35–27** from generating the DSC Input Resume signal. If this signal is not generated, the computer cannot be interrupted to transfer the second word from C2 into the variable memory.

Whenever a tape read operation is initiated, the complete record must be read in order to allow a longitudinal parity check to be performed utilizing the check mark found at the end of each record. However, the computer may require less words for the input operation than are found in the record being sensed. In this case, the tape unit operation is maintained until the check mark at the end of the record is gated into D register for transfer to the L register. However, all tape characters in excess of those required by the computer cannot be gated from C2 register into the variable memory because Terminate **FF33–34** is set. The line pulse generator and the D sequence control continue to operate in all other respects until the last tape data character of the record is sent to the D register and subsequently gated to L register by the signal from **Ā38–46. Ā38–52** is subsequently enabled to clear the line pulse generator **FF38–10** in the normal fashion. The next character to be sensed following the end of record is the check mark which follows the last data character at approximately a 135 microsecond interval. Therefore, it is entered into D register more than

75 microseconds after FF**38–10** has been cleared. FF**38–62** is consequently set to indicate the end of the record, and a positive pulse via N**38–65** is applied to the 100 microsecond delay **38–66** to insure that flip-flop **38–10** is subsequently set even though the check mark may contain all 0's. The setting of FF**38–10** again causes the line pulse generator counter flip-flops to advance, which allows the check mark in D register to be gated into the L register by the output of $\overline{A}$**38–46**. $\overline{A}$**38–52** is enabled in the normal fashion which clears FF**38–10**. The output from $\overline{A}$**38–52** is also supplied via N**38–54** to condition $\overline{A}$**38–55** to sense the contents of L register, inasmuch as a positive signal is now generated from the set output terminal of End Of Record FF**38–62**. If the L register at this time does not contain all 0's, then a Longitudinal Parity Error signal is generated. Furthermore, the output from $\overline{A}$**38–52** is supplied to $\overline{A}$**38–56** which in turn stops the tape unit. When the tape unit stops, a Tape Out Drop Go signal is applied to set the Drop Go flip-flop F**2–13** in the F**2** register to subsequently clear F**2–14**.

3.7. *Subcommand and Memory Access Control*

The memory access control of FIGURE 39 provides a variety of control signals relating to the overlap referencing of the permanent and variable memories and the arithmetic unit as regards withdrawal of instruction and operand data words. Unless otherwise specified, the term "permanent memory" hereafter includes the twistor input switch, since both utilize the W and Z registers. In addition, circuits are provided for testing the legality of the addresses contained in U***10–23**. Flip-flops **39–10** and **39–11**, when set, respectively indicate a referencing of the permanent or variable memories for an operand data word held at an address indicated by the contents of U***10–23**. Flip-flop **39–12** is set when U***10–23** indicates that the operand is held in the arithmetic unit. Flip-flops **39–13** and **39–14**, when set, respectively indicate the referencing of the permanent or variable memories by the address contained in the P register. If FF**39–13** is set, then $\overline{A}$**39–25** is enabled at MP10 to transfer the instruction from Z register to U register. Likewise, if FF**39–14** is set, then $\overline{A}$**39–26** is enabled to transfer the instruction from O register into U register. At best, only one of the operand flip-flops and one of the instruction flip-flops may be set for the duration of any machine cycle. If both the P register and U***10–23** attempt to reference the same memory (other than an "A" address) during a machine cycle, the operand flip-flop alone is set since the execution of a current instruction takes precedence over the acquisition of the next instruction from memory.

All flip-flops **39–10**, **39–11**, **39–13**, and **39–14** are unconditionally cleared at the end of a machine cycle by MP15 via N**39–22**. Flip-flop **39–12** is unconditionally cleared at the beginning of a machine cycle at MP0 time via N**39–24**. Flip-flop **39–10** may be set at MP2 time by a significant negative output from $\overline{A}$**39–15** which is responsive to the presence of positive signals respectively indicating the legality of the address held in U***10–23**, the absence of a V or A operand address, and the presence of an Operand Op Code in the current instruction held in U**0–5**. The flip-flop **39–11** may similarly be set by an output from $\overline{A}$**39–16** at MP2 time if the current instruction requires an operand for execution and if the U***10–23** register stages hold an address of the variable memory. Flip-flop **39–12** is set by a significant negative output at MP3 time from $\overline{A}$**39–17** if the address in U***10–23** is 03000 and if the Op Code of the current instruction, as presently existing in U**0–5**, requires an operand for execution of the instruction.

Flip-flop **38–13** may be set by either one of two signals respectively emanating from $\overline{A}$**39–18** and **39–19**. $\overline{A}$**39–18** is responsive at MP1 to an indication from the P register that it holds a permanent memory address, as well as to an indication that the current instruction does not require a jump in the program sequence. $\overline{A}$**39–19** is responsive at MP2 time to an indication that there is an instruction jump to be made, as well as to a legal permanent memory address (other than TIS) in U***10–23**. This is because a jump instruction requires that the next instruction following be taken from a memory location indicated by its indexed "T" address portion. However although flip-flop **39–13** may be initially set at the beginning of a machine cycle, any subsequent setting of flip-flop **39–10** by $\overline{A}$**39–15** clears FF**39–13** so that referencing of the permanent memory for an operand takes priority over referencing of the same memory for the next instruction. Similarly, flip-flop **39–14** may be set by either one of two signals coming from $\overline{A}$**39–20** or $\overline{A}$**39–21**, but only if the system is not operating in the Real Time Mode. $\overline{A}$**39–20** is responsive only to an indication that P register contains a variable memory address, as well as to the absence of an instruction jump. $\overline{A}$**39–21** requires the presence of a jump instruction as well as the presence of a variable memory address within U***10–23**. Again, any subsequent setting of V Operand flip-flop **39–11** by $\overline{A}$**39–16** at MP2 automatically clears flip-flop **39–14** if the latter had also been set at MP1. One other means for clearing flip-flops **39–13** and **39–14** is an output from $\overline{A}$**39–23** which responds at MP3 time to the grant of an input-output data transfer or interrupt operation in order that a special instruction can be forced into U register instead of the next instruction of the main program.

In the event that an attempt is made during the same machine cycle to simultaneously reference the same memory for both the next instruction and a data operand, the acquisition of the latter takes preference such that referencing of the memory for the next instruction cannot occur until the machine cycle immediately following. The address in the P register of the next instruction cannot be incremented by 1 until after it has been sent to the memory address register for use in referencing the memory. Flip-flop **39–31** controls the normal incrementing of the address in the P register by a value of 1. When FF**39–31** is in a clear condition, the positive output from its clear terminal enables $\overline{A}$ gates **39–36** through **39–39** to respectively issue time sequenced signals Clear P*, P to P*, and P*+n To P at MP3 through 6. These are sent to the program address counter in order to increment the address in the P register in the manner previously described. PAC Add FF**7–112** is normally clear at these times so that 1 is added to the value. However, upon FF**39–31** being set, the negative signal from its clear output terminal prevents generation of the four above described signals, FF**39–31** is normally cleared unconditionally at MP0 time, but may be set in a variety of ways. One way is by a significant negative output from $\overline{A}$**39–28** which responds to the setting of the V Operand flip-flop **39–11** by $\overline{A}$**39–16** as well as to an indication that P register contains a variable memory address. Additionally, an output from $\overline{A}$**39–30** sets FF**39–31** if the W operand flip-flop **39–10** is set by $\overline{A}$**39–15** together with indications that P register holds a permanent memory address. Also, FF**39–31** must be set during execution of certain specific instructions even though overlap would seem possible. Any negative input signal to $\overline{O}$**39–33** enables $\overline{A}$**39–32** at MP2 time to set FF**39–31**. Some of these signals are generated during execution times of a Repeat instruction **40** and the repeated instruction. Others are generated at the grant of a data transfer or interrupt operation, or in the event that an illegal address is contained in U***10–23**. Additionally, FF**39–31** may be set if the P address is illegal.

The circuits for determining the memory to which the U***10–23** address refers will now be considered, since these form a portion of the memory access control unit of FIGURE 39. Since an "A" operand address is represented by the octal value 03000, all three of the U* register stages **10**, **11**, and **12** are cleared as has previously been explained. Thus, the two positive signals from the clear output terminals of U*10 and U*11 are applied to Ā39–43, and the negative signal from the set side of U*12 is inverted via N39–42 and also applied to Ā39–43. Ā39–43 thereby generates a significant negative signal which, when inverted via N39–44, indicates an "A" operand address. If the U* register contains an operand address in the variable memory, then both U*10 and U*11 are cleared while U*12 is set, as has previously been described. This condition of these three U* stages indicates that the address is in the range from 04000 to 07777. Ā39–40 is thus responsive to the cleared conditions of U*10 and U*11 and the set condition of U*12 to generate a significant negative output which in turn is inverted by N39–41 and used as a positive signal enable for other circuits. If a binary 1 value resides in either or both U*10 and U*11, then the highest order octal digit of the U* register address must be 1 or above, thus indicating a location in the permanent memory. Ā39–45 generates a positive output unless both U*10 and U*11 contain binary 0's, as evidenced by positive signals from their clear output terminals.

A positive output from Ō39–48 indicates that a jump is to be made in the instruction program. For the Sub-Routine Jump and Unconditional Jump Instructions, as well as the interrupt instruction, the jump always occurs and a negative signal is applied to Ō39–48 via N39–49. During execution of the Index Jump Instruction however, the condition under which the jump is taken is that the number contained in the R memory location, as specified by the "*j*" portion of the index jump instruction, must not equal —1. If $R_j \neq -1$ then the input to N30–46 is negative which in turn applies a positive signal to Ā39–47.

Ā39–55 generates a signal P To W–1 at MP1 time if no jump is to be executed and if the P address is one in the permanent memory without there also being a permanent memory address held in U*10–23. Thus, the contents of the P register may be gated into the W address register to initiate an instruction reference at this time. In like fashion, a signal P to V–1 is generated from Ā39–56 at MP1 time in the absence of a jump request and V memory address in U*10–23, if the P register holds a V address. Furthermore, in the absence of a current instruction calling for an operand or a jump, the P register address may be used to reference the permanent or variable memory regardless of the value contained in U*10–23. This function is provided by Ā gates 39–57 and 39–58, which respectively generate the signals P To W–2 and P To V–2. These signals, however, are inhibited by a negative output from N39–53 which in turn requires a positive signal from Ō39–51. The output from Ō39–51 is positive if a jump is requested, or if the instruction held in the U register requires an operand from one of the memories for its execution.

In the event that the address in U*10–23 is used to reference the variable or permanent memories, Ā39–59 and 39–60 respectively provide the signals U* To W and U* To V. The signals are generated at MP1 time if the U*10–23 stages have a W or V address. and if the instruction Op Code calls for either an operand or a jump function. The positive output from Ō39–51 is used to indicate either of the latter two conditions.

Circuitry is further provided in FIGURE 39 to determine the legality of the address held by U*10–23. For example, a significant negative output is generated from Ā39–81 at MP3 time if it receives a positive input from Ō39–80. Four negative inputs can be provided to Ō39–80 any of which will cause the Illegal U*10–23 Address Error signal to be generated. One of these inputs is generated by Ā39–79 if the U*10–23 address references the accumulator (an "A" address) for an operand when the current instruction is one for which this is prohibited. Such instructions may be ascertained by reference to FIGURE 9 wherein it is seen that instructions 20, 21, 22, 44, and 45 will generate a positive signal Illegal A Operand Address applied to one input Ā39–79. There are also instructions for which an operand should never be acquired from the permanent memory. These instructions are any of those in Table 2 having a higher order octal digit of 2, plus instructions 44 and 45. Ā39–78 detects the presence of any one of these instructions in the U register, as well as an indication that the U*10–23 register contains a permanent memory address.

Another of the conditions which generates an output from Ā39–81 is that occurring when the instruction requires an operand, but the U*10–23 register contains an address in the range from 34XXX to 37717, for which there are no permanent memory locations. This range of addresses is detected by the following circuitry. Ā39–62 is responsive to 1 bits in U*10, U*11, and U*12 to generate a significant negative signal if the U* address has values of either 34XXX, 35XXX, 36XXX, or 37XXX. This is so, since U*10 and 11 contain binary bits comprising the highest order octal digit, while U*12 contains the binary order bit of value 4 for the second highest order octal digit. Ā39–69 is responsive to 1 bits in all of the U* stages 13–17 so that a significant negative output is generated therefrom if the U* address has the values X37XX or X77XX. Ō39–67 is responsive to a binary 1 bit in either or both of U* stages 18 and 19 so as to generate a significant positive output if the U* address has the values of XXX2X, XXX3X, XXX4X, XXX5X, XXX6X, or XXX7X. By inverting the output of Ā39–62 and Ā39–69 and applying same to Ā39–66 along with the output from Ā39–67, it is therefore seen that a significant negative output from Ā–66 indicates that the U* address is in the range of from 3772X to 37777, which are the address locations in the twistor input switch. If the U* address is in this range, the negative output from A39–66 will disable any significant negative output from Ā39–65. However, if the address range of the U* register stages is between 34XXX to 37717, there is a positive output from Ā39–66 together with a significant negative output from Ā39–62 which thereby enables Ā39–65 to generate an output indicating that the U* address refers to a non-existent memory location. Ā39–77 in turn is responsive to the negative output from Ā39–65 and to the command Operand Instruction.

The fourth input to Ō39–80 is derived from Ā39–82 if during the execution of jump instruction there is a positive signal from Ō39–83. Ō39–83 in turn generates this signal if the U* address is in the range of from 34XXX to 37777 because of inputs derived from both Ā39–65 and Ā39–66. There are no instructions stored in the TIS. Furthermore, since there are no instructions stored in the arithmetic unit an "A" address is also illegal during jump. Furthermore, it is illegal to acquire instructions from the variable memory if the computer is operating in the Real Time Mode, with this condition being detected by Ā39–84. It should be noted that the output from Ā39–82 is also applied to Ō39–33 thus setting the Normal Advance PAC Lockout flip-flop 39–31 in order to maintain the illegal address in the P register to which it is also transferred during the jump execution.

In connection with FIGURE 39, it should further be noted that the negative output from Ā39–65 is applied to Ā39–15 to prevent the W Operand flip-flop 3–10 from being set. Also, Ō39–74 is responsive to either an illegal address 34XXX to 37717 or a TIS address to disable Ā39–19 from setting the W instruction FF39–13.

FIGURE 40 shows details of the skip and arithmetic lockout control which may come into play to modify the overlap operation when either an extended sequence operation is being performed in the arithmetic unit, or an arithmetic Skip or Selective Sense instruction is to be executed. As mentioned before, an extended sequence instruction is one which requires more than one machine cycle time in which to be executed. These instructions in the present computer repertoire are Multiply 14, Divide 15, Square Root 16, and Shift 42. During the first execution cycle for any one of these instructions, an Extended Sequence flip-flop within the Arithmetic Sequence Control unit is set and remains so until the operation is completed. This flip-flop is shown and described in the above identified co-pending U.S. application Serial No. 137,805. However, as noted previously, execution of non-arithmetic instructions can continue. Any arithmetic instruction subsequently reference from one of the memories, during the time that an extended sequence operation is being currently performed by the arithmetic sequence control, causes termination of the instruction reference cycles since all instructions must be executed in the order acquired and such arithmetic instruction cannot be executed in the arithmetic unit until the current extended sequence operation is completed. In FIGURE 40, the Arithmetic Lockout flip-flop **40–10** is unconditionally cleared at MP3 time via **N40–41** and remains in this condition unless subsequently set by one of several conditions. The setting of flip-flop **40–10** is due to a significant negative output from **Ā40–11** at MP12 time if a positive output exists from **Ō40–12**. The set condition of the Extended Sequence flip-flop applies a positive signal to **Ā40–13** and **Ā40–14** such that if either an arithmetic instruction or a Load R/Store R instruction (if it requires an "A" operand from the arithmetic unit) is placed into the U register, then a signal will be generated which sets **FF40–10**. A signal is also applied to **Ō40–12** by **Ā40–15** if a Select Sense instruction is detected from **U*0–5** at the time that an extended sequence is being performed. Besides setting **FF40–10**, the output from **Ā40–11** also causes the transfer of the address in P* register into P register so that the arithmetic or Load R/Store R instruction will be acquired again during the next machine cycle, at which time the extended sequence may be terminated so that this instruction can be performed in the arithmetic unit. The setting of **FF40–10** also causes a positive signal to be applied to **Ā40–18** such that a Clear **U0–9** is generated from **Ō40–19** at MP14 time to clear these U register stages so that the instruction will not be transferred to U* and interference with the extended sequence execution. **Ō40–19** also receives another Clear **U0–9** signal from FIGURE 9 which is generated if **U0–5** contains an illegal Op Code. **Ō40–20** is further responsive to the signals Clear **U0–23** —1 and Clear **U0–23** —2, both from FIGURE 41 which are simultaneously applied to **Ō40–19** to insure that the complete U register is cleared. **Ā40–22** is provided to stop the computer if an appropriate match is detected in **U10–23**, but this function is delayed until completion of an extended sequence operation.

If a Selective Skip instruction 41 is placed into the U register, **FF40–24** is set at MP4 time if a match is detected between a depressed control panel button and a binary 1 bit in one of the U register stages **17–23**. The setting of **FF40–24** generates a positive signal to Ā40–27 which is gated at MP14 time and causes the generation of a significant positive output from **Ō40–34**. This output is applied via N40–36 to set FF Skip flip-flop **40–35**, and is further applied to Ā40–37 to clear the P register in the event that a positive signal is generated from Ō40–40. However, a positive output is obtained from **Ō40–40** only in the event that the system is either operating in a repeat mode, or if the Normal Advance PAC FF39–31 is set. If a positive output is obtained from Ō40–40, the setting of Skip flip-flop **40–35** also causes a signal P*+*n* To P to be generated from **Ā40–13** at the following MP15 time in order to calculate the address of the instruction following the skipped instruction. A positive output from **FF40–35** is always gated through **Ā40–39** at MP15 time to clear the Next Instruction Lockout flip-flop if such has been set due to repeat operation.

Four other conditions cause the generation of a positive output from **Ō40–34** in FIGURE 40. These conditions may or may not occur for any of the four skip instructions 10, 11, 12, or 13 shown in Table 2. For example, if the instruction Equal skip 10 is being executed, the operand specified by the address portion of this instruction is subtracted from an operand held in the arithmetic unit to determine if the difference is 0, which in turn indicates equality therebetween. Thus, after the subtraction in the arithmetic unit has occurred, a test is made on the difference and if this is 0, a positive Zero Test Enable signal is applied to Ā gate **40–31** which is conditioned by the command Equal Skip. On the other hand, if the difference is not equal to 0, then the negative Zero test Enable signal is inverted and applied to **Ā40–30** also responsive to the command Non-Equal Skip. The other two skip instructions, Less Skip 12 and Greater Equal Skip 13 are also executed in the arithmetic unit by substracting the memory operand from the accumulator operand and determining the sign of the difference. The signal Skip Enable is positive if the accumulator operand is greater or equal to the memory operand, and negative if the memory operand is greater than the accumulator operand. Ā gates **40–28** and **40–29** respectively respond to the commands Greater Skip and Less Skip in order to determine if the skip conditions are met.

FIGURE 41 shows the Repeat Control Circuitry which is utilized when executing both the Repeat Instruction 40, and the repeated instruction which is acquired immediately thereafter. The Repeat Mode flip-flop **41–10** is set at MP3 time of the machine cycle following that in which Repeat instruction 40 is acquired. The setting of **FF41–10** produces a positive signal at its set output terminal and a negative signal from its clear output terminal, with the latter causing the periodic setting of the Normal Advance PAC Lockout flip-flop **39–31** via **Ō39–33**. Additionally, a Set R Memory Add Flip-Flop signal is generated by **Ā41–14** at the following MP7 time. Following the transfer of the Repeat instruction from the U register to U* register at MP4 time, the Next Instruction Lockout flip-flop **41–12** is set at MP12 time by **Ā41–13**. This will make negative the signal P To V Or W Enable to prevent the translation of the P register address. Also, **U0–23** will no longer be unconditionally cleared by **Ā41–24**, so that the repeated instruction is retained in U register until it has been executed the required number of times.

The instruction acquired after the Repeat instruction 40 is the instruction to be repeated a number of times according to a value held in the R memory address location 7. This value is normally represented by a negative number. Each time the repeated instruction is executed, a 1 is added thereto. When the R memory value eventually equals —1, the execution of the repeated instruction ceases, and the next instruction acquired from memory. The Terminate Repeat flip-flop **41–16** will thus be set a significant negative output from **Ā41–17**, which in turn responds to an indication that R equals —1 from FIGURE 16 at MP8 time during the repeat mode operation. The setting of **FF41–16** in turn causes the signal **P*** To **P** to be generated at the following MP13 time from **Ā41–18** and **Ō41–19**, which transfers into the P register the address of the next instruction following the repeated instruction. The positive output from **FF41–16** is also gated at MP13 time to reset **FF41–12** in order that the P counter address may be translated. The negative output from the clear terminal of **FF41–16** is applied via **Ō41–21** to **Ā41–22** to reset the Repeat Mode flip-flop **41–10** in the next following MP0 time. MP3 unconditionally clears the Terminate Repeat flip-flop **41–16** via

N41–23. Ō**41–19** is also responsive to Ā**8–81** to transfer P* to P when an illegal Op Code is detected in **U0–5.**

If the circuitry of FIGURE 42 requests the system to execute either a data transfer or an interrupt sub-routine during the time that a repeat mode operation is being performed, the Interrupt Repeat flip-flop **41–32** is set via Ā**41–33**, which also clears FF**41–12** so that the P register address can be translated to acquire from memory the instructions of the sub-routine. The setting of FF**41–32** thereby disables an output from Ā**41–16**, since after execution of an input-output data transfer or an interrupt sub-routine, the repeat mode must be resumed to continue the execution of the repeated instruction until the number in R7 is equal to −1. If Ā**41–33** clears FF**41–12** prior to the clearing of the Repeat instruction 40 from U***0–5**, FF**41–12** will be set again at MP12 time by Ā**41–13**. In order to insure that FF**41–12** remains clear for the duration of the interrupt operation, the positive signal from FF**41–32** is applied via Ā**41–34** at MP13 time to again clear FF**41–12**, after which time this flip-flop remains clear. The negative signal from the clear terminal of FF**41–32** is also applied via Ō**41–21** and Ā**41–22** to clear FF**41–10**. After the data transfer or interrupt has been terminated, provision is made to again withdraw the same Repeat instruction 40 from memory at which time the Repeat flip-flop **41–10** is again set and the execution of the repeated instruction continued until terminated by the value of **R7**=−1.

In the event that the instruction following a Repeat instruction **40** is either another Repeat instruction 40 or an instruction which is not repeatable, then the repeat operation must be terminated and an error indicated. The first Repeat instruction **40** acquired from memory is eventually placed in the U* register, and the instruction following is placed in U register. Thus, if the instruction following is also a Repeat instruction **40**, Ā**41–26** senses this condition from both the U* and U command translator circuits at MP12 time to set Illegal Repeat flip-flop **41–25** which thereafter clears **U0–23** by Ā**41–27**. At the same time, the output signal from Ā**41–26** is applied via Ō**41–28** to indicate an error, with this output further being directed via **N41–30** to set the Terminate Repeat flip-flop **41–16** whereby the repeat mode will be terminated in the manner previously described. Another way of indicating an Illegal Repeat instructioin error and terminating the repeat mode is if the instruction following the Repeat instruction 40 is one which cannot be repeated (other than a Repeat instruction 40). All instructions which should not be repeated cause the generation from FIGURE 9 of an Illegal Repeat command which is applied to Ā**41–29**. This Ā gate is also responsive to the presence of the Repeat instruction 40 in the U* register, as well as to the absence of the Interrupt Repeat signal, to generate an output signal via Ō**41–28** which indicates error and also sets Terminate Repeat flip-flop **41–16**. The instruction in **U0–23** is now performed only once in the usual fashion.

3.8. *Interrupt and Data Transfer Control*

FIGURE 42 shows the priority circuits whereby signals are generated interrupting the main program of the computer so that either an input-output data transfer may be accomplished using the peripheral equipment, or an interrupt sub-routine may be entered for any one of several reasons. The execution of certain types of interrupt conditions have priority over others in the event that two or more different kinds of interrupts are requested simultaneously by the system. The following Table 8 indicates the relative priority of the requests.

TABLE 8

(1) IN/OUT Data Transfer Channel **2**
(2) IN/OUT Data Transfer Channel **1**
(3) Error Interrupt
(4) Start Pulse Interrupt
(5) IN/OUT Channel **2** Interrupt
(6) IN/OUT Channel **1** Interrupt or External Unit Interrupt

The priority indicated in Table 8 is performed by the circuitry of FIGURE 42 in the following manner. Request flip-flops **42–10** through **42–15** may be respectively set by a signal individually requesting an operation of Table 8. For example, FF**42–10** is responsive to a signal requesting an Input-Output Channel **2** Data Transfer. When once a Request flip-flop **42–10** through **42–15** is set, it will remain in this condition until cleared by a signal indicating that its request has been granted by the system. The set output terminals of flip-flops **42–10** through **42–15** are respectively applied to the group of Ā gates **42–22** through **42–27**, which in turn are connected to the set input terminals of the Priority (1 through 6) flip-flops **42–16** through **42–21**. In addition, the clear output terminals of Request flip-flops **42–10** through **42–15** are connected to those Ā gates of the group **42–22** through **42–27** which are associated with the Request flip-flops having lower priority than its own. For example, the clear output terminal of flip-flop **42–10**, which is responsive to the highest priority signal Request Input-Output Data Transfer Channel **2**, is connected directly to Ā gates **42–23**, **42–24**, and via Ā**42–28** and **N42–29** to the inputs of Ā gates **42–25** through **42–27**. In similar fashion, the clear output terminal of Request FF**42–11** is connected directly to Ā**42–24** and through Ā**42–28** and **N42–29** to Ā gates **42–25** through **42–27**. The clear output terminals of FF**42–12** through **42–14** are likewise connected to Ā gates associated with the lower priority flip-flops.

The above described arrangement of interconnections provides for the determination of priority between the input signals applied to Request flip-flops **42–10** through **42–15**. For example, assuming initially that all flip-flops **42–10** through **42–15** are cleared, a signal Request Input-Output Data Transfer Channel **2** from FIGURE 33 will set FF**42–10**, which in turn applies a positive signal to Ā**42–22** so that at the following MP0 time, Priority 1 flip-flop **43** is set. However, as long as Request FF**42–10** is set, none of the other Ā gates **42–23** to **42–27** can be enabled to set their corresponding priority flip-flops **42–17** through **42–21**. This is so, because the negative signal from the clear terminal of FF**42–10** insures that positive output signals will be generated from Ā gates **42–23** and **42–24** no matter what the polarity of the other input signals thereto. In addition, Ā**42–28** maintains a positive signal therefrom which, when inverted via **N42–29**, insures that Ā gates **42–25** through **42–27** also generate positive signals no matter what the values of their other input signals. Conversely, if Request flip-flop **42–12** is the first of the group to be set, then it disables only Ā gates **42–25** through **42–27**, while enabling Ā gate **42–24** to set the Priority 3 flip-flop. Although the setting of FF**42–12** does not disable Ā gates **42–22** and **42–23**, thus allowing the Priority 1 or Priority 2 flip-flops to be set during a time that Priority 3 is set, this is permissible. Therefore, the setting of any of the Request flip-flops **42–10** through **42–15** sets its own associated Priority flip-flop **41–16** through **42–21**, but prevents setting of any of the Priority flip-flops associated with a lower priority input signal.

The set outputs from Priority flip-flops **42–16** through **42–21** are in turn gated at MP3 time via a set of Ā gates **42–30** through **42–35**, whose output signals are utilized to reset the corresponding Request flip-flops **42–10** through **42–15**. Therefore, only when its associated priority flip-flop is set will the Request flip-flop be cleared. The priority flip-flops **42–16** through **42–21** are in turn unconditionally cleared at the following MP12 time via **N42–36**.

The setting of either the Priority 1 or Priority 2 flip-flops **42–16** or **42–17** indicates that an Input-Output

Transfer of a data word or words must be made on channel 1 or channel 2 respectively. Thus, the negative signal from the clear terminal is applied via **Ō42–37** to produce a positive signal Data Transfer. In turn, the output from **Ō42–37** is applied via **N42–38** to **Ō42–39** to provide the signal Data Transfer Interrupt. In like fashion, **Ō42–39** is also responsive to an input signal indicating that an Interrupt operation is granted, said input signal coming from **Ō42–40** via **N42–41**. As may be seen in FIGURE 42, **Ō42–40** is responsive to any negative signal from the clear output terminals of flip-flops **42–18** through **42–21**. The output from **N42–41** is also applied via **N42–42** to generate a positive signal Interrupt.

Whenever an interrupt sub-routine not a data transfer is to be performed, the execution of the main program is interrupted for a minimum of six machine cycles, during which time the granting of any other interrupt must be disabled. This six machine cycle lock-out for interrupt is performed by four flip-flops **42–45** through **42–47**. Normally, these four flip-flops are in their clear conditions such that positive signals from their clear output terminals are applied to **Ā42–55**, which is thereby enabled at MP1 time to generate a negative output inverted by **N42–56** and applied to the set of AND gates **42–24** through **42–27**. If any one of the Request flip-flops **42–12** through **42–15** is set by the time this gate signal from **N42–56** arrives, then one of the associated gates **42–24** through **42–27** will be enabled to set the corresponding Priority flip-flops **42–18** through **42–21**. Of course, if two or more of the Request flip-flops **42–12** through **42–15** are set, then only the highest interrupt priority flip-flop is set. When one such interrupt priority flip-flop is set, a positive signal is generated from **N42–42** to enable **Ā42–43** to generate a negative signal at the next following MP3 time, which in turn sets **FF42–44**. The setting of **FF42–44** now supplies a negative signal input to **Ā42–55** thus disabling it from generating a significant negative output at the next MP1 time. Consequently, no positive gating signal can be generated from **N42–56** as long as **FF42–44** is in its set condition. In like fashion, a negative signal from the clear output terminal of any of the other flip-flops **42–45** through **42–47** results in a disabling of gates **42–24** through **42–27**.

The sequence setting and clearing flip-flops **42–44** through **42–47** is shown in Table 9, where the MP times of setting and clearing are given for each of the Lockout Enable 1, 2, 3, and 4 flip-flops in FIGURE 42.

TABLE 9

CYCLE #	MP Time	FF 1	FF 2	FF 3	FF 4
	1 3	CLEAR	CLEAR	CLEAR	CLEAR SET
1	1 15		SET	SET	
2	13	SET			
3	11				CLEAR
4	9			CLEAR	
5	7		CLEAR		
6	5	CLEAR			

As previously mentioned, the setting of any one of the Priority 3 to 6 flip-flops **42–18** through **42–21** at MP1 time causes the setting at MP3 time of flip-flop **42–44**. Therefore, at MP1 time of the next machine cycle, **Ā42–55** is disabled from passing the MP1 pulse, so that none of the Ā gates **2–24** through **42–27** can be sampled at this time. However, MP1 is applied to **Ā42–18** to which is applied the positive signal from the set output terminal of **FF42–44**. Thus, **FF42–45** (**FF3** in Table 9) is set at this time. The positive signal output from it is applied to **Ā42–49** which is sampled at MP15 time of the first lockout cycle so as set **FF42–46** (**FF2** in Table 9).

During the second lockout cycle, the MP13 pulse gates at **Ā42–50** with the positive signal from the set output terminal of **FF42–46** to set **FF42–47**. The positive output from the set terminal of **42–47** (**FF1** in Table 9) is applied to **Ā42–51** whose output in turn is connected to the clear input terminal of **FF42–44**. Thus, at MP7 of the third lockout cycle, **FF4** is cleared by an output from **Ā42–51**. However, **Ā42–55** remains disabled due to the fact that one or more of the remaining flip-flops **FF3**, **FF2**, and **FF1** is still in its set condition. Therefore, the MP1 pulse still may not be gated to Ā gates **42–24** through **42–27**. At machine cycle number **4**, an **MP9** pulse switches at **Ā42–52** with the positive signal now coming from the clear output terminal **FF4** to clear **FF3**. During cycle **5** an **MP7** pulse is gated through **Ā42–53** to clear **FF2**, while during the sixth lockout cycle, an MP5 pulse is gated through **Ā42–54** to clear **FF1** (**FF42–7**). Therefore, it is not until **MP1** time of the machine cycle following the sixth lockout cycle that a sampling signal may be generated from **Ā42–55**, which in turn will sample the contents of flip-flops **42–12** through **42–15** to determine if an interrupt request has been set therein during execution of the preceding interrupt grant. If so, then one of the Priority flip-flops **42–18** through **42–21** may be set to initiate execution of another interrupt, at which time the operation of flip-flops **42–44** through **42–47** will again cause a six machine cycle lockout as described above.

Flip-flop **42–57**, when set, indicates that the computer is in its Real Time mode. **FF42–57** may be set from **Ā42–58** at MP10 time upon the grant of a start pulse interrupt as indicated by the setting of **FF42–19**, and by the presence of a positive signal External Force Real Time.

FIGURE 43 discloses circuitry responsive to the grant of a data transfer or interrupt for forcing particular values into certain portions of the U register. Ā gates **43–10** through **43–18** respectively generate set signals which are applied to the indicated U register stages in order to set same to values indicated in Table **10** below.

TABLE 10

Priority	Set U0–5 (Octal)	Set U6–9 (Decimal)	Set U10–23 (Octal)
1	45	10	---------
2	46	8	---------
3	46	15	10000
4	46	14	10010
5	46	14	10020
6	46	14	10030

To illustrate the setting of the U register according to Table 10 given above, assume that the operation of Data Input Output Transfer in Channel **2** (priority number 1) is granted in **FIGURE 42**. The positive signals Data Transfer, Data Transfer/Interrupt, and Not Data Transfer 1 are generated from **FIGURE** 42 and applied to circuitry in **FIGURE** 43. Thus, AND gates **43–10**, **43–11**, and **43–15** are enabled at MP10 time to generate significant negative outputs which set various ones of the U register. **Ā43–10** sets **U0** and **U3** to place the octal operation code 45 therein. Ā gates **43–11** and **43–15** respectively set **U6** and **U8** to place a binary value 1010 into the index portion of the U register which has a decimal equivalent of **10**. There is no forced setting of **U10–23** for either a Channel **1** or Channel **2** Data Transfer operation. In somewhat similar fashion, if a Data transfer is requested and granted for channel **1**, then only Ā gates **43–10** and **43–11** are energized inasmuch as the signal $\overline{\text{Data Transfer 1}}$ is now negative. Thus, the same operation code 45 is placed in **U0–5** but only a single stage U6 is set to place a decimal value of 8 into **U6–9**.

Assuming now that an Error Interrupt request is granted in FIGURE 42, the signals Data Transfer/Interrupt, Error Interrupt, and Interrupt are all positive to enable Ā gates **43–11**, **43–13**, **43–15** (because $\overline{\text{Data Transfer 1}}$ is positive) **43–14** and **43–16** at MP10 time. Ā**43–14** sets **U0**, U3 and U4 to force the octal operation code 46 into **U0–5**. In addition, Ā**43–16** sets U7 and **U11**, Ā**43–15** sets U8, while Ā**43–11** sets U6 to force binary 1's into all of the U6–9 registers to thereby indicate a decimal value of 15. In addition, the setting of **U11** from Ā**43–16** forces an address into **U10–23** having the octal value of 10000. However, if a start Pulse Interrupt is granted in FIGURE 43, then the following operation is observed. The Start Pulse Interrupt signal is negative but the signals Interrupt, $\overline{\text{Data Transfer 1}}$, and Data Transfer/Interrupt are positive. The Interrupt signal sets U7 via Ā**43–16** and further sets the operation code 46 into **U0–5** via Ā**43–14**. The Data Transfer Interrupt signal sets **U6** and **U8**. However, for this kind of interrupt, U9 remains cleared so that the decimal value held by **U6–9** is 14 instead of 15, as was the case for an Error Interrupt. Furthermore, both **U11** and **U20** are set, the latter from Ā**43–17** in response to an output from Ō**43–19**. Therefore, the octal address held in **U10–23** is 10010. Similarly, a grant of a Channel **2** Interrupt provides for forcing the operation code 46 and the decimal value 14 into **U0–5** and **U6–9** respectively, as well as the setting of **U11** and **U19**. The latter two settings therefore cause **U10–23** to contain an octal address value of 10020. The grant of a Channel **1** Interrupt priority sets both **U19** and **U20** to force the address 10030 into **U10–23**, besides setting **U0–5** to 46 and **U6–9** to 14.

FIGURE 44 shows circuitry for generating the Request Error Interrupt and Request External interrupt signals for use in FIGURE 42. If the E register, not forming a part of the present invention, determines that an error has occurred for which a sub-routine should be initiated, a positive signal applied to Ā**44–19** is gated through at MP15 to set **FF42–12**. Also, if **FF44–10** is set by a start pulse from an external unit, the positive signal from the set terminal of **FF44–10** is gated via Ā**44–11** at MP10 time to set **FF44–12**. The positive signal from **FF44–12** is gated by Ā**44–13** to generate a Request Start pulse Interrupt which in turn sets **FF42–13**. A signal from Ā**44–13** also is applied to clear **FF44–10**. However, External Stop Pulses flip-flop **44–12** cannot be cleared until the Start Pulse Interrupt is granted as indicated by the setting of **FF42–19**, which in turn will generate a Start Pulse Interrupt grant signal from Ā**42–33** which is applied to the clear terminal of **FF44–12**. The output from Ā**44–13** also is applied to **N44–14** for transfer to the E register.

In the event that a External Interrupt signal is received from an external unit, **FF44–15** is set which in turn sets **FF44–17** at the next following MP11 time because of Ā**44–16**. The set side of **FF44–17** is applied to Ā**44–18** to enable same at MP15 to generate a Request External Interrupt signal applied to **FF42–15**. In turn, the output from Ā**44–18** clears **FF44–15**, with **FF44–17** remaining set until the External Interrupt Request is granted in FIGURE 42. The use of the flip-flop and gates in FIGURE 44 is primarily for the purpose of synchronizing the computer and the external units.

OPERATION

FIGURES 45 through 55 are timing diagrams which illustrate the operation of the present invention under various conditions next to be described. In these figures, the set condition of a control flip-flop is represented by an "up" waveform. The content of a register during any machine cycle or portion thereof is indicated by an appropriate word or words (sometimes abbreviated) placed beneath an "up" waveform. When the register is empty, the waveform is "down." In any machine cycle, the instruction acquired from memory and placed in U register is referred to as NI (Next Instruction), while the instruction held in U* register which is being executed is CI (Current Instruction). Thus, NI subsequently becomes CI. The digit or digits following NI and CI normally indicate the sequential nature of the memory address locations in which the instruction words are stored. For example, **NI45** is acquired from a memory location having an address 1 higher in value than the address of N144.

4.0. *Overlap*

FIGURE 45 is a timing diagram showing the operation of the invention during several machine cycles when different memories are referenced for the instruction, and operand. The assumption is made that the P register contains an address location in in the variable memory at the beginning of the first machine cycle under consideration and that there is no instruction within the instruction register U or U* at commencement of this first machine cycle. At MP0 time, machine cycle **1**, registers Z, O, W, and V are unconditionally cleared. This signal is followed at MP1 time by signals from both Ā**39–56** and Ā**39–58** either one of which transfers the contents of P register to V register. Outputs are obtained from both of these gates under the assumption that P register contains a variable memory address, together with the absence of an operand Op Code in **U0–5** and the absence of a W address in U***10–23**. V register now contains the address of **NI1**. The V reference flip-flop **23–10** is also set at MP1 time so that the location in variable memory corresponding to the address held in the V register may be read from and written into during machine cycle **1**. A signal is also generated from Ā**39–20** to set the V Instruction flip-flops **39–14** to allow the next instruction to be gated from the O register into the U register later during the machine cycle. At MP2 time, the WX shaper flip-flop **27–10** is unconditionally set, while at MP3 time both the V Read flip-flop **23–13** and the WY shaper flip-flop **27–11** are unconditionally set. The setting of the V Read flip-flop causes the variable memory to be referenced and **NI1** withdrawn therefrom and placed into the O register. Ā**39–36** also clears **P*** register at MP3 time inasmuch as the normal advance PAC lock out flip-flop **39–31** is cleared.

At MP4 time of machine cycle **1**, the R register is unconditionally cleared, and Ā**39–37** transfers **P** register to P* register. P* register now contains the address of **NI1**. Likewise at M4 time, **U0–5** is transferred to U***0–5** unconditionally, but this does not change U***0–5** inasmuch as **U0–5** is assumed to be cleared at the beginning of the machine cycle. Additionally, Ā**14–40** clears U*6–9 which was also initially assumed to be clear at the beginning of machine cycle **1**. At MP5 time, Ā**39–38** clears P register of its **NI1** address. At MP6 time, Ā**39–39** transfers the contents of P* through the address counter, where it is incremented by 1, and into P register such that **P** register now contains the address of **NI2** having a value of 1 higher than the address of

NI1. R_j to R FF16–36 is set at MP5 time and attempts to reference the R memory. However, inasmuch as U*6–9 is cleared, no location of the R memory is sampled and R register thus remains empty. Flip-flop 16–36 is unconditionally cleared at MP7 time.

At MP9, Ā41–24 clears U0–23 because of the cleared condition of the Next Instruction flip-flop 41–12. At MP10, the V Write flip-flop 23–24 is set in order that NI1 can be read back into its variable memory location and thus retained for future use. Ā39–26 transfers the contents of the O register into U0–23 such that the U register now contains NI1. At MP12 time, Ā14–29 transfers the "j" indexing address from U6–9 to U*6–9 so that the latter now contains the "j" address portion of the current instruction CI1 to be executed in the next following machine cycle 2. R register is also unconditionally cleared at MP12 time. At MP13 time, the R_j To R flip-flop 16–36 is unconditionally set to reference the R memory at the location indicated by the address now held in U*6–9 to withdraw the contents of R_j and place in the R register. R register now contains the index number of the current instruction which will be used to modify the "S" address portion of NI1 to form the "T" address portion of CI1. U10–23 is also unconditionally gated to U*10–23 at this time. At MP14 time of machine cycle 1, the V write flip-flop 23–24 is reset, thus ending the memory cycle for the variable memory. At MP15 time, the V instruction flip-flop 39–14 is reset as well as flip-flop 16–36.

Assume now that the address contained in the P register, that of NI2, still references a location in the variable memory. Also assume that the "S" address portion of NI1 identifies a location in the permanent memory. At MP0 of machine cycle 2, Ā13–95 generates a signal if the Op Code portion of NI1 held in U0–5 is one calling for an index operation. Thus, the contents of U10–23, holding the "S" address portion of NI1, is added to the contents of R register with the sum being placed into U*10–23. The "S" address contents of the latter register is now replaced by the "T" address portion of the current instruction CI1, with this address also being assumed to reference a location in the permanent memory. Also occurring at MP0 time of machine cycle 2 is an unconditioned clearing of the Z, O, W, and V registers.

At MP1 time, Ā39–56 transfers the contents of P register to V register, such that the latter now contains the address of NI2. Also at MP1 time, Ā39–59 transfers U*10–23 to W register so that a reference may be had of the permanent memory to withdraw the operand utilized during the execution of the CI1. Ā18–94 sets R To OR Gates flip-flop 18–93 in order that the indexing number held in R register may be returned to the R memory where it is retained for future use. Also at MP1 time of machine cycle 2, Ā39–20 sets the V Instruction flip-flop 39–14. At MP2 time, Ā39–15 sets the W Operand FF39–10, while the WX Shaper FF27–10 is likewise set.

At MP3 time, the V Read FF23–13 is set to enable transfer of NI2 into O register. The WY Shaper FF27–11 is also set at MP3 time. Also Ā39–36 clears P* of its NI1 address so that at MP4 time, Ā39–37 may transfer P into P* with the latter now containing the address of NI2. Also at MP4, the R register is unconditionally cleared of the CI1 index value which has been transferred back to R memory via the OR gates whose flip-flop 18–93 is also cleared. In addition, U*6–9 stages are cleared by the signal from Ā14–40 so that the CI1 "j" index address portion is lost. At MP4 time, U0–5 is unconditionally gated to U*0–5, such that the latter now contains the operation code of CI1. However, U0–5 is not cleared at this time, so that the CI1 operation code is now contained in both U0–5 and U*0–5 stages where it may be translated by each of the command translator circuits associated therewith.

At MP5, Ā39–38 clears the P register of its NI2 address. Also, FF16–36 is set so that an attempt is made to reference the R memory. However, by MP5 U*6–9 stages have been cleared of any address so that there is no actual withdrawal of information from the R memory at this time. Therefore, R register remains empty. At MP6, Ā39–39 transfers the contents of P* register to the P register (via the address counter where 1 is added thereto) such that the address of NI3 is now placed in P register. It is assumed for purposes of this description that P register now holds an address indicating a location in the permanent memory. At MP8 of machine cycle 2, the permanent memory strobe FF27–12 is set to gate the CI1 operand into the Z register. The contents of Z register may now be gated to the arithmetic unit by circuitry not shown where it is used in the execution of CI1 according to the Op Code held in U*0–5.

Ā41–24 clears U0–23 stages at MP9 times so that at MP10 time, Ā39–26 gates the contents of O register into U register, with the latter now containing NI2. O register continues to hold NI2 so that said instruction may be written back into the variable memory by the setting of the V Write flip-flop 23–24.

At MP12, Ā14–29 gates the "j" address portion of NI2 held in U6–9 into U*6–9 where it becomes the index address portion of CI2. At MP13, FF16–36 is set to gate the contents of R memory (at the address held in U*6–9) into the R register for use in modifying the "S" address portion of NI2 presently held in U10–23 (and also in U*10–23 by this time). At MP15, FF39–14 and FF39–10 are unconditionally cleared as well as FF16–36.

During machine cycle 3, CI2 will be executed and an attempt will be made to reference the memory for the next instruction NI3 whose address is now contained in the P register. At MP0 time the Z, O, W, and V registers are unconditionally cleared. Furthermore, the Op Code contained in U0–5, which is that of NI2, is assumed to require an indexing operation such that Ā13–95 adds the contents of U10–23 to R register and places the sum in U*10–23. Thus, U*10–23 now contains the "T" address portion of CI2, which is assumed to refer to a location in the variable memory. The P register, it will be remembered, now contains the address of NI3 which is assumed to be located in the permanent memory. Thus, at MP1, Ā39–55 transfer P into W register while Ā39–60 transfers U*10–23 into the V register. At the same time, Ā39–18 sets the W Instruction FF39–13 so that NI3 can subsequently be gated from Z register into U register. At MP2, Ā39–16 sets V Operand FF39–11 indicating that the CI2 operand is to be referenced from the variable memory according to the CI2 "T" address portion now held in the V register.

At MP3 of machine cycle 3, Ā39–36 again clears P* preparatory to transfer of P into P* at MP4 time. At MP4, Ā14–40 clears U*6–9, while Ā39–37 gates P into P* so that the latter now contains the address of NI3. U0–5 is also gated unconditionally to U*0–5 at MP4 time such that the latter now contains the operation code for the CI2 which is to be executed during machine cycle 3. transfers P* to P by way of the address counter which At MP5, P register is cleared by Ā39–38. At MP6, time Ā39–39 transfers P* to P by way of the address counter which increments the NI3 address to the NI4 address. Also, during the set time of the V Read FF23–13 the CI2 operand is withdrawn from the variable memory and placed in O register for eventual transfer to the arithmetic unit where it is operated upon in accordance with the CI2 Op Code. At MP8, the permanent memory is strobed by FF27–12 to withdraw NI3 therefrom and place same in the Z register.

At MP9, Ā41–24 clears U0–23, and at MP10 time Ā39–25 transfers Z register contents into U0–23. U0–23 now contains NI3 which will be executed as CI3 during machine cycle 4 next to be described. At MP12, Ā14–29 transfers U6–9 to U*6–9 such that the "j" index address portion of CI3 is now contained therein. At MP13, FF**16–36** is set to reference the R memory in accordance with this address so that the index value is transferred into the R register. At MP15, FF**39–13** and FF**39–11** are reset, as well as FF**16–36**.

At MP0 time of machine cycle **4**, Z, O, W, and V registers are again unconditionally cleared. Assuming that the operation code of CI3 calls for an indexing operation, Ā**13–95** adds U**10–23** to R register and places the result in U***10–23**. It is now assumed for purposes of this description that the CI3 "T" address portion held in U***10–3** refers to the arithmetic unit as the location of the operand to be used in the execution of CI3 during machine cycle **4**. P register at the beginning of machine cycle **4** is assumed to contain a permanent memory address for NI4. Therefore, at MP1 Ā**39–55** transfers P into W register while Ā**39–18** sets the W Instruction of FF**39–13**. However, since U***10–23** now holds an "A" address, there is no gating of U***10–23** into either the permanent or variable memory address registers. Instead, Ā**39–17** sets A Operand FF39–12 at MP3 time which indicates to the arithmetic unit that the operand may be found therein. Also during MP3 time, Ā**39–36** clears P* which is filled with the P address at MP4 time by action of Ā**39–37**. U***6–9** is also cleared at MP4 by Ā**14–40**. At MP5 P register is cleared and the incremented P* address transferred thereto at MP6 time so that the address of NI5 is calculated. Also at MP4 time **U0–5** is transferred to U***0–5** so that the latter now contains the operation code for CI3, with said operation code still being maintained in **U0–5** until MP9 time.

At MP9, Ā**41–44** clears U**0–23** which is subsequently filled at MP10 time by NI4 from Z register. At MP12, the "*j*" index address portion from **U6–9** is gated to **U*6–9** for use in withdrawing the modifying index number from the R memory and placing same in the R register.

FIGURE 45 therefore illustrates the overlap operation of instruction, acquisition and execution where different memories are simultaneously referenced to supply both the next instruction and the operand of the current instruction. This is the normal operation of the system in the absence of certain special instructions, subsequently to be described, which will modify the overlap function. It is therefore seen that the effective processing time of most arithmetic instructions is but one machine cycle, inasmuch as the cycle required for the acquisition of the instruction is performed during the same machine cycle that the preceding instruction is being executed. Furthermore, part of the indexing function for the current instruction is performed in the machine cycle during which it is acquired from memory, thus further increasing system speed. It should be appreciated in connection with FIGURE 45 that the W and V addresses selected for the instructions and operands are purely arbitrary and serve as an example designed to aid in the understanding of the overlap circuitry previously described in detail.

4.1. *Overlap inhibition*

FIGURE 46 is a timing diagram showing the operation of the invention if both the P register and the "T" address portion of the current instruction refer to an address in the same memory. In such a situation, the acquisition of the current instruction operand (if such is required by the Op Code) takes precedence over the acquisition of the next instruction with the latter function being postponed until the next following machine cycle. In order to provide a certain degree of continuity in the explanation of the different overlap operations and modifications thereof, it is assumed for purposes of this discussion that FIGURE 46 commences with machine cycle **5** immediately following machine cycle **4** shown in FIGURE 45. Thus, U*–**23** contains NI4 which is executed as CI4 during machine cycle **5** of FIGURE 46. P register also contains the address of NI5 which is now assumed to reference the permanent memory as does also the "S" address portion of NI4.

At MP0 time of machine cycle **5** in FIGURE 46, the Z, O, W, and V registers are cleared. Also, under the assumption that the NI4 Op Code in **U0–5** requires an indexing operation, Ā**13–94** adds the contents of **U10–23** to R register and places the sum in **U*10–23**, with the latter now containing the "T" address portion of CI4 assumed to indicate a location in the permanent memory. At MP1, Ā**39–59** transfers **U*10–23** to the W register where it is used to reference the permanent memory to withdraw the CI4 operand later during the cycle. It will be noted that this time however, that neither Ā**39–55** nor Ā**39–57** can generate a signal to transfer the contents of P register into W register, since negative signals are generated from both **N39–54** and **N39–53** because of the presence of a W address in **U*10–23** and an operand code in **U0–5**, respectively. These conditions, however, do not prevent Ā**39–18** from setting W Instruction **FF39–13** at MP1. However, at MP2, Ā**39–15** sets the W Operand **FF39–10** and in so doing clears W Instruction **FF39–13** to prevent any entry into the U register from the Z register at MP10 time during machine cycle **5**. Also at MP2, Ā**39–30** sets the Normal Advance PAC **FF39–31** which in turn prevents the incrementing of the P register address. This is so because no signals are generated from Ā gates **39–36** through **39–39** at MP times 3, 4, 5, and 6, respectively. Thus, P register continues to store the NI5 address while P* register continues to hold the NI4 address. The NI5 address in P register will therefore be available at the beginning of machine cycle **6** to reference the permanent memory and withdraw NI5.

As may be observed in FIGURE 46, the remaining operations during machine cycle **5** as regards acquisition of the CI4 operand are similar to those described in FIGURE 45. For example, Ā**41–24** clears **U0–23** at MP9, but there is no subsequent placing of new information in this register at **MP10** time inasmuch as NI5 is not acquired during this machine cycle. The CI4 operand is withdrawn from the permanent memory at **MP8** time and placed in Z register, but since W Instruction **FF39–13** is cleared, no gating is permitted between the Z register and the U register. At MP12 in machine cycle **5**, Ā14–29 transfers the empty contents of **U6–9** to **U*6–9**, with both thereafter remaining clear at the end of this signal. The R register is unconditionally cleared at MP12 and an R memory strobe is attempted at MP13 by the setting of **FF16–36**. However, at this time **U*6–10** is empty so that a location in R memory cannot be referenced. Thus, no information is transferred into the R register. At MP15, W Operand **FF39–10** is cleared.

During machine cycle **6**, there is no current instruction to be executed inasmuch as no instruction was withdrawn from memory in the preceding machine cycle **5**. Therefore, during machine cycle **6**, the only function of the program system is to withdraw NI5 from the memory and begin the indexing operation thereof if such is required. At MP0 time, Ā13–95 generates a signal to add the contents of **U10–23** to R register, not withstanding the fact that there is no operation code residing in **U0–5**. This is so, since Ō9–22 generates a negative output except when an operation code specifically prohibiting an index function resides in **U0–5**. However, since **U10–25** and R register are both empty at MP0 time of machine cycle **6**, **U*10–23** continues to remain empty even after this addition occurs. Since P register contains a permanent memory address, that of NI5, Ā39–18 sets the W Instruction FF39–13 at MP1. Also, both Ā**39–55** and Ā**39–57** generate signals either of which transfers the P register into the W register for use in referencing the permanent memory. At MP3, 4, 5, and 6, Ā gates **39–36**, **39–37** and **39–38** and **39–39** respectively generate signals to increment the P register address by 1 so that the address of NI6 is calculated and placed in P register by MP6 time. These four gates are enabled during machine cycle **6** inasmuch as the Normal Advance PAC **FF39–31** is unconditionally cleared at MP0 time. Also at MP4 time, U***0–5** is set with the contents of **U0–5**. However, **U0–5** at this time is empty such that U***0–5** is emptied of the CI5 Op Code. At MP10, $\overline{A}$**41–24** gates the Z register into the U register, the latter now containing NI5. At MP12, $\overline{A}$**14–29** gates the NI5 "j" address portion to U***6–9** where it thereby becomes the index address for CI5. This address references R memory and withdraws the index number therefrom for placement in the R register at MP13 time. Therefore, **NI5** will be executed during the next following machine cycle **7** as **CI5**, and an attempt will be made to reference a memory for **NI6** whose address is held in the **P** register. If different memories can be referenced, or if no operand is required for execution of CI5, then an overlap operation will occur similar to that described in FIGURE 45.

4.2. *Extended sequence*

FIGURE 47 illustrates the operation of the invention in the event that the arithmetic unit is executing an extended sequence instruction requiring more than one machine cycle. As noted previously, most instructions using the arithmetic unit require only one five microsecond machine cycle for execution. However, the multiply, divide, square root, and shift instructions in most cases require more than one machine cycle except where certain short cut or error conditions may be present. These instructions are known as extended sequence instructions. Normally, non-arithmetic instructions may be processed by the present invention even during execution of an extended sequence instruction, since the ASC unit necessary for execution of an extended sequence instruction is independent of the instruction register and command translator circuits in the program unit. Therefore, after initiation of an extended sequence operation, which may occupy from two to thirteen machine cycles, non-arithmetic instructions are continuously withdrawn from the memories by the program unit, either in over-lap fashion as indicated in FIGURE 45 or in fashion of FIGURE 46, and executed by parts of the system other than the arithmetic unit. However, since the arithmetic unit in the present system is able to execute only one arithmetic instruction at a time, the acquisition of an arithmetic instruction during an extended sequence requires that the instruction referencing be discontinued until this arithmetic instruction can be accepted by the arithmetic unit.

Again for the purpose of continuity in this description, FIGURE 47 is shown as beginning with machine cycle **7** which is assumed to follow machine cycle **6** of FIGURE 46. Therefore, the U register is shown to contain NI5 while the **P** register holds the address of NI6 calculated during machine cycle **6**. Likewise, U***6–9** holds the "j" address portion of CI5. It is further assumed that CI5 calls for one of the four extended sequence operations which will require several machine cycles, beginning with machine cycle **7**, for execution in the arithmetic unit. As noted in machine cycle **6** of FIGURE 46, the "S" address portion of NI5 is considered to reference a location in the variable memory, while the P register holds a permanent memory address. At MP0 time of machine cycle **7** in FIGURE 47, $\overline{A}$**13–95** again adds the contents of **U10–23** to R register and stores the resulting "T" address portion of CI5 in U***10–23**. It is here assumed that the "T" address references the variable memory so that there will be no inhibiting of the overlap operation during machine cycle **7**. Therefore, at MP1, $\overline{A}$**39–55** transfers P register into W register, while $\overline{A}$**39–60** transfers U***10–23** into the V register. $\overline{A}$**39–18** also sets the W Instruction **FF39–13** at this time. At MP2, $\overline{A}$**39–16** sets the V Operand **FF39–11**. This circuitry enables the transfer of the acquired NI6 from the Z register into the instruction U register later during the machine cycle **7**, as well transfer to the arithmetic unit of the CI5 operand from the O register.

Since there is no setting of the Normal Advance PAC **FF39–31**, $\overline{A}$ gates **39–36** through **39–39** are enabled during MP times 3, 4, 5, and 6 to increment by 1 the address in the P register, so that the latter now contains the address of NI7. Also, the CI5 operand is withdrawn from variable memory and placed into O register where it will be subsequently transmitted to the arithmetic unit and operated upon in accordance with the CI5 Op Code held both in U***0–5** and **U0–5**. The variable memory control flip-flops are not shown in FIGURE 47 since this operation is similar to that shown in FIGURE 45.

During machine cycle **7**, a signal is generated from the appropriate one of $\overline{A}$ gates **9–20**, **9–19**, **12–34**, or **12–36** to set an Extended Sequence flip-flop in the arithmetic unit, which will remain set until execution of the instruction is completed. Therefore, a positive signal is applied to both $\overline{A}$ gates **40–13** and **40–14** for this period of time. Since the Op Code for CI5 is also contained in **U0–5** and represented as the NI5 Op Code, the Arithmetic Instruction signal to $\overline{A}$**40–13** is also positive during machine cycle **7** such that $\overline{A}$ gate **40–13** emits a significant negative output to $\overline{O}$**40–12** which in turn supplies a positive output to $\overline{A}$**40–11**. However, $\overline{A}$**40–11** is not gated until MP12 time. Thus, since the U register is cleared at MP9 time by an output from $\overline{A}$**41–24**, there is no chance that the extended sequence arithmetic instruction itself will cause the setting of the Arithmetic Lockout flip-flop **40–10**. Only the first arithmetic instruction following the extended sequence instruction will cause the setting of this flip-flop in FIGURE 40.

As mentioned above, the U register is cleared at MP9 time and the Z register transferred thereto at MP10 by virtue of $\overline{A}$**39–5**. The U register now contains **NI6**, which is assumed to be a non-arithmetic instruction not requiring use of the arithmetic unit for execution. The index number is withdrawn from the R memory at MP13 time in accordance with the "j" address held in U***6–9**. Therefore, at the end of machine cycle **7**, execution of CI5 has been initiated and NI6 acquired from memory which in turn will be executed during machine cycle **8** by a portion of the system other than the arithmetic unit. It is also assumed that the "S" address portion of NI6 refers to a variable memory location, whereas the P register indicates a permanent memory location for NI7.

During machine cycle **8**, NI6 will be executed as CI6, and an attempt will be made to acquire NI7 unless the same memory must be referenced simultaneously by both U***10–23** and the P register. After indexing it is assumed that the CI6 "T" address portion placed in U***10–23** still requires referencing of the variable memory. Therefore, as may be seen in FIGURE 47, P register is gated to the W register while U***10–23** is gated to the V register, with the W Instruction **FF39–13** and the V Operand **FF39–10** both being set. Additionally, the address in P register is incremented by 1 during MP3 to MP6 times such that it now contains the address of **NI8**. The CI6 operand is withdrawn from the variable memory and placed into the O register for use by the system in accordance with the CI6 Op Code residing in both **U0–5** and U***0–5** by MP4 time of machine cycle **8**. Additionally, at MP9 time U register is cleared and NI7 transferred thereto from the Z register. Thus, the functions performed by the system in machine cycle **8** of FIGURE 47 are exactly the same as performed in machine cycle **3** of FIGURE 45 as regards acquisition of the next instruction and execution of the current instruction held in the instruction register. The only difference is that the arithmetic unit is independently executing the previous extended sequence instruction CI5 as indicated by the condition of the Extended Sequence flip-flop. The operation of the arithmetic sequence controls in the arithmetic unit, however, do not interfere with execution of the non-arithmetic instruction CI**6** by other parts of the system.

In order to illustrate how the overlap operation can be inhibited during execution of an extended sequence instruction by the arithmetic unit, it is assumed that NI7 is a non-arithmetic instruction whose "S" address portion refers to the variable memory. In machine cycle **9**, NI7 will be executed as CI7 since it is an instruction not calling for use of the arithmetic unit. In addition, **P** register will attempt to reference the permanent memory to acquire NI**8**. Assuming that CI7 requires indexing, this is performed at MP0 time of machine cycle **9** and the resulting "T" address portion of CI7 placed into U***10–23**. However, although the NI7 "S" address is one in the variable memory, it is here assumed that after indexing the CI7 "T" address refers to the permanent memory. Therefore, P register is inhibited from acquiring **NI8** since acquisition of a current instruction operand takes priority over acquisition of the next instruction when both are located in the same memory. At MP1 time of machine cycle **9** **Ā39–59** transfers U***10–23** (since CI7 calls for an operand) to the W register for use in withdrawing the CI7 operand and placement thereof in the Z register. No gating signal is generated at MP1 time for transferring P into the W register. However, **Ā39–18** does set W Instruction flip-flop **39–13** at MP1 time, with FF**39–13** subsequently being cleared at MP2 time by a generation of a signal from **Ā39–15** to set W Operand FF**39–10**. In addition, the signal at MP2 time from **Ā39–15** also cause **Ā39–30** to set the Normal Advance PAC **FF39–31** to prevent the **P** register address from being incremented. Therefore, P register continues to hold the address of **NI8** during machine cycle **9** so that said instruction may be acquired during the next machine cycle **10**. Since there is no withdrawal of the next instruction during machine cycle **9**, the U register, when cleared at MP9 time, remains empty so that there can be no retreaval of an index number from the R memory at MP13 time. The operation of the program unit during machine cycle **9** is therefore identical to the function performed by it during machine cycle **5** of FIGURE 46, except that the arithmetic sequence control in the arithmetic unit is at this time continuing the execution of the extended sequence instruction CI5 initiated in machine cycle **7**.

During machine cycle **10**, there will be no current instruction present in the U* register so that the only function of the program unit is to acquire NI8 from the permanent memory according to the address in **P** register, which remained unchanged during machine cycle **9**. This function is performed in exactly the same manner as indicated in machine cycle **6** of FIGURE 46 such that at MP10 time, NI8 is transferred from Z register into U register. Also, since the Normal Advance PAC FF**39–31** is reset at MP0 time of machine cycle **10**, the address in P register is incremented by 1 so that the address of NI9 resides therein by MP6 time.

It is now assumed that NI8 is an arithmetic instruction such that its Op Code residing in **U0–5** causes a positive Arithmetic Instruction signal to be applied to **Ā40–13** by MP10 time of machine cycle **10**. If the Extended Sequence flip-flop remains set, since execution of the extended sequence instruction has not yet been terminated, **Ā40–13** supplies a positive signal to **Ā40–11** via **Ō40–12** at time MP12 of machine cycle **10**. The output from **Ā40–11** sets the Arithmetic Lockout **FF40–10** which is unconditionally cleared at the next following MP3 time of machine cycle **11**. In addition, the positive output from **Ō40–12** is applied through **NI4–30** to disable **Ā14–29** from gating **U6–9** to U***6–9** at MP12 time. Instead, the positive **Ō40–12** output is applied to **Ā14–38** to insure that U***6–9** is cleared by MP12 time of machine cycle **10**. Additionally, the significant negative output from **Ā40–11** causes **Ō41–19** to transfer the contents of P* register directly to P register, thus erasing the NI9 address in P register and replacing it with the NI8 address of P* register. There is no need to first clear P register prior to this transfer, inasmuch as double gating is used between P* and P as previously explained. At MP14 time of machine cycle **10**, the set condition of **FF40–10** enables **Ā40–18** to clear U register stages **0–9** via **Ō40–19**. Thus, at the end of machine cycle **10**, P register again contains the address of NI8, while **U10–23** is filled with the S address portion of NI8. **U0–9** is empty.

At the beginning of machine cycle **11**, **U0–5** is empty to add the contents of **U10–23** to the contents of R register. However, R register is at this time empty since the empty condition of **U*6–9** during MP13 time of machine cycle **10** prevented the referencing of **R** memory. Thus, the sum is equal to the "S" address portion of NI**8**. If it is assumed that this CI8 "S" address portion refers to an address in permanent memory, then it will be impossible for **Ā39–55** to generate a P To W signal at MP1 time. However, since U0–5 is now empty during MP1 time of machine cycle **11**, the Operand Instruction signal from FIGURE 9 is negative such that the output from **Ō39–51** is also negative to thereby enable **Ā39–57** to generate a P To W signal which transfers the P register into W register for use in again acquiring **NI8** from the permanent memory. On the other hand, if the CI8 "S" address portion refers to a variable memory location, then both **Ā39–55** and **Ā39–57** will generate a signal either of which causes the P register transfer into W register. In no case is U***10–23** transferred to either W or V register at this time, since as previously mentioned, there is no positive operand instruction signal generated from FIGURE 9.

Since the Normal Advance PAC **FF39–31** remains cleared during machine cycle **11**, the P register address will again be incremented by 1 during MP times 3 through 6 such that the address of the NI9 is calculated. At MP10 time of machine cycle **11** (not shown in FIGURE 47), NI8 will again be transferred from the Z register to the U register exactly as shown in machine cycle **10**. If, at MP12 time of machine cycle **11**, the Extended Sequence flip-flop continues to be set, then P* register will again be transferred to P register so that NI8 will again be withdrawn from permanent memory during machine cycle **12** time. If, however, the Extended Sequence flip-flop is cleared by MP12 time of machine cycle **11**, thus indicating that execution of the extended sequence instruction CI5 has been or will shortly be terminated, then the Arithmetic Lockout **FF40–10** cannot be set and NI8 remains in U register where it will be indexed and executed as CI8 during the next following machine cycle **12**. Therefore, as long as the Extended Sequence flip-flop remains set, the program unit operates in accordance with the functions indicated in machine cycles **10** and **11**, after which the arithmetic instruction NI8 may be executed by the arithmetic unit and the instruction sequencing for NI9 continued.

It should also here be mentioned that the instruction Load R/Store R, if requiring an operand from the arithmetic unit, is also considered to be an arithmetic instruction such that **Ā40–14** sets Arithmetic Lockout flip-flop **40–10**. **FF40–10** is unconditionally cleared at the following MP3 time of the next cycle in order to allow the arithmetic instruction to again be withdrawn from memory for comparison against the Extended Sequence flip-flop to determine if the execution of the extended sequence instruction is yet completed. This operation is performed during each machine cycle until the arithmetic unit is available to execute the next following arithmetic instruction.

4.3. *Instruction jump*

FIGURE 48 is a timing diagram illustrating the operation of the invention when either an Unconditional Jump, Sub-Routine Jump or Index Jump instruction enters the instruction register. In the case of the Unconditional Jump, the "T" address portion of this instruction is employed to reference the memory for the next instruction, and not the address held in the P register. Thus, the next instruction to be acquired from memory may have an address other than the one sequentially following the address location from which the Unconditional Jump instruction is taken. The execution of a Sub-Routine Jump instruction is similar to the Unconditional Jump Instruction in that its "S" address portion (this instruction is not indexable) is utilzied to reference memory for the next instruction). However, the instruction address currently held in the P register is stored in the R memory so that upon completion of the sub-routine, the program can revert to this instruction, which is held in a memory location having an address sequentially following the address of the Sub-Routine Jump instruction. For the Index Jump instruction, its "S" address portion is used to reference memory only if the number held in R memory, at a location indicated by the "*j*" address portion of the jump instruction, is not equal to —1. At the same time, a value of 1 is added to the index number which is thereupon placed back into R memory for examination at some future time by another Index Jump instruction. If the R memory index number is equal to —1, then no jump in the program is made and the P register is used to acquire the next instruction following the index jump instruction.

In FIGURE 48, it is assumed for purposes of this discussion that during a machine cycle **25**, the next instruction acquired during the cycle is an Unconditional Jump instruction **NI15** from permanent memory which is placed in the U register at MP10. It is further assumed that the **CI14** being executed during machine cycle **25** requires an operand from variable memory such that overlap occurred during machine cycle **25** in a manner theretofore described. Therefore, at MP8 time of machine cycle **25**, the following conditions are assumed to be present: **U*10–23** contains the **CI14** "T" address while **U*0–5** contains the **CI14** Op Code. The W register holds the address of **NI15** which is placed into the U register at MP10 time. The P register holds the address of **NI16** which was computed during MP3 through MP6 times of machine cycle **25**. **P*** register continues to hold the address of **NI15**. The V register is assumed to hold the "T" address portion of **CI14** used to address the variable memory and acquire the **CI14** operand now located in O register.

As stated above, **NI15**, which is an Unconditional Jump Instruction, has been placed in U register by MP10 time of machine cycle **25**. This instruction is indexable so that an index number is withdrawn from the R memory and added to the "S" address portion with the sum being placed into **U*10–23** register as the "T" address portion of **CI15**. The manner in which this indexing is accomplished has been described previously in connection with the other timing diagrams. During machine cycle **26** the "T" address portion of **CI15** will be used to reference one of the memories for the next instruction to be executed following execution of **CI15**. It is assumed that the "T" address portion of **CI15** refers to a location in the permanent memory. P register also refers to a location in a permanent memory having an address one higher than the address from which **NI15** was withdrawn. However, the "T" address portion of **CI15** may refer to any location in permanent memory or it could just as well refer to a location in the variable memory if the system is not operating in the Guide Mode. In FIGURE 39, the signal Interrupt, SRJ, UCJ from the command translator circuits of **U0–5** is positive because of the unconditional Jump code residing therein. Therefore, the negative output from **N39–50** disables **Ā39–55** from generating the signal P To W. In like fashion, the negative signal from **N39–50** enables **Ō39–51** to generate a positive signal which in turn disables **Ā39–57** from generating a signal P To W. Therefore, it is not possible to transfer the contents of P register to W register during MP1 time of machine cycle **26**. However, **Ā39–59** generates a signal U* To W which does transfer the "T" address portion of **CI15** to the W register where it will be used to address the permanent memory to acquire an instruction therefrom which is placed in the U register via the Z register. At MP2 time of machine cycle **26**, **Ā39–19** sets the W Instruction flip-flop because of the jump command, while **Ā5–31** clears the P register of the **NI16** address. At MP3 time, **Ā5–25** also transfers the contents of **U*10–23** into the P register, while **Ā39–36** clears the P* register in preparation for transfer of P to P* occurring at MP4 time.

Regarding now the address held in both the W register and the P register at conclusion of MP3 time, it is assumed for purposes of this discussion that this address refers to a location in the permanent memory wherein is contained the **NI3** instruction previously acquired during the machine cycle **3** shown in FIGURE 45. Obviously, the **CI15** "T" address portion could refer to other instruction addresses anywhere within the permanent or variable memories.

Returning now to machine cycle **26** of FIGURE 48, at MP4 time **Ā39–37** gates P to P* such that P* now contains the address of **NI3**. At MP5, **Ā39–36** clears P. At MP6 time, **Ā39–39** gates P* through the address counter to P, with the latter now holding the address of **NI4**. At MP9, U register is cleared and at MP10, **Ā39–25** gates Z register to U register with the latter now holding **NI3**. This instruction will be executed during machine cycle **27** in a manner similar to that shown in FIGURE 45. Thus, in recapitulation, the address of the instruction to be acquired following the execution of the unconditional jump instruction is the "T" address portion of the unconditional jump instruction and not the number contained in the P register. Furthermore, P and P* registers must also be filled with this new address so that the normal instruction sequencing operation continues for succeeding machine cycles beginning with the address of NI.

FIGURE 48 also shows machine cycles **36** and **37** wherein it is assumed that acquisition and execution of a Sub-Routine Jump instruction occurs. For purposes of this discussion, it is assumed that **NI28** is a Sub-Routine Jump instruction and has been acquired from memory and placed in U register by MP10 time of the cycle. It is further assumed that **CI27** is being executed during machine cycle **36** that **U*10–23** contains the "T" address portion of **CI27**. Another assumption is that the addresses for the **CI27** operand and **NI28** refer to different memories so that an overlap operation has occurred during machine cycle **36**. Thus, P* register holds the **NI28** address, P register holds the **NI29** address, while **U*10–23** contains a variable memory address for the operand of **CI27**.

Although the "S" address portion of a Sub-Routine Jump instruction is not indexable, an R memory reference must still be made during MP12 and 13 times in order that the contents of P register can be read into R memory at a location indicated by the "*j*" address portion of **JI28**. Thus, **Ā14–29** gates **U6–9** at MP12 time of machine cycle **36** so that at MP13 time, the content of RJ is placed in the R register. However, at MP15 time, **Ā16–34** now clears R register. This is so since a positive signal from the command translator circuits of **U0–5** is applied to **Ā16–34** indicating the presence of a Sub-Routine Jump Op Code therein, and it is assumed for purposes of this description that there is no skip condition present. At MP0 time of machine cycle **37**, **Ā16–25** transfers the contents of P register into R register, with the latter being placed into RJ during MP1 through MP3 time of machine cycle **37**. It is thus seen that for a Sub-Routine Jump instruction, the address of the next instruction residing in P register at the beginning of a machine cycle will be placed into the R memory and there kept until the subroutine is terminated. The last instruction of a subroutine program might be an Unconditional Jump instruction having an "S" address portion equal to 0, and "*j*" address portion for referencing R memory at this location. Therefore, the NI29 address would be acquired, added to 0, and placed in U***10–23** from whence it would be used to reference memory in order that the main program may be reentered at the point from which the jump occurred.

Returning now to machine cycle **27** of FIGURE 48, the S address portion of CI**28** residing in U***10–23** must be transferred to one of the memory address registers to withdraw the first instruction of a subroutine program. Assume for purposes of this description that the CI**28** "S" address portion references the variable memory. In this case, **Ā39–60** transfers U***10–23** to V register. There is no transfer of the P register to the W register at this time due to the disabling of both Ā gates **39–55** and **39–57** because of the presence of the subroutine jump signal applied to Ō gate **39–48** via N**39–49**. The V register now contains the "S" address portion of the subroutine jump instruction CI**28** which is considered in the present example as the address of NI**60**. **Ā39–21** also sets V Instruction FF**39–14** at MP2 time of machine cycle **37** so that NI**60** may be gated from O register to U register at MP10 time. Additionally, **Ā5–31** clears P register at MP2, while **Ā5–25** transfers U***10–23** into P register at MP3. The P register now holds the address of NI**60** which is actually the "S" address portion of CI**28**. At MP3, P* is cleared and P transferred to P* register at MP4 in the normal manner. **Ā39–38** now clears P at MP5 while **Ā39–39** transfers P* to P register via the address counter so that P register now contains the address of NI**61**, which is the second instruction of the sub-routine program. At MP9 time, U register is cleared and **Ā39–26** transfers O register to U register at MP10. All of the instructions comprising the sub-program routine are now withdrawn from sequential address locations memory and executed in a manner identical to withdrawal and execution of the main program. There may or may not be overlap during the sub-routine program.

Machine cycles **50** and **51** of FIGURE 48 illustrate the acquition and execution of an Index Jump instruction. Assume that NI**40** is acquired during machine cycle **50**. Further assume that it is an Index Jump instruction having an "S" address portion referring to a permanent memory location. Also assume that the operand of CI**39**, which is executed during machine cycle **50**, refers to a variable memory location so that overlap has occurred during machine cycle **50** and the P register now contains the address of NI**41**.

Although the "S" address portion of an Index Jump instruction is not indexable, reference must be had to the R memory at a location indicated by the "*j*" address portion in order to determine if the number there residing is equal to —1. This operation is initiated at MP12 time of machine cycle **50** by the transfer of U6–9 to U*6–9 enabled by **Ā14–29**. At MP13, the contents of RJ are entered into R register as shown in FIGURE 48. If this number is not equal to —1, i.e., one or more of the R register stages contains a binary 0, then the signal R=—1 generated from **Ā16–44** will be positive. This positive signal is inverted and then applied to N**39–46** for inversion again before application to **Ā39–47**. The presence of an Index Jump Op Code in U register also causes a positive signal to be applied to the other input of **Ā39–47**, whose negative output thus indicates that the jump is to be performed. During machine cycle **50**, it is assumed that the contents of R does not equal —1 so that a jump will be performed during machine cycle **51** in order to acquire an instruction from memory having an address other than the one held by P register. At MP15 of machine cycle **50**, **Ā40–42** sets the R Memory Add flip-flop **17–52** since it is assumed that no skip operation is present. The setting of FF**17–52** will thereby add 1 to the contents of R register as it is being transferred back into the R memory during MP1 through MP3 time of machine cycle **51**.

Since it is assumed that the "S" address portion of CI**40** (the index jump instruction) references a permanent memory location, then U***10–23** is transferred to W register by **Ā39–59** at MP1. **Ā39–19** also sets the W Instruction flip-flop **39–13** at MP3 so that the next instruction may be transferred from Z register into U register during MP10 time. At MP2, **Ā5–31** clears P register, and **Ā5–25** transfers **10–23** into P register at MP3. Both P register and W register now contain the "S" address portion of CI**40** which is here assumed to be the address location of NI**90**. In order that P register will be equipped to reference memory for NI**91** during machine cycle **52**, it is now necessary to clear P*, transfer P to P*, and increment P* by 1 upon its being gated back to P. Therefore, by MP6 time of machine cycle **51**, P register contains the address of NI**91**. At MP10, NI**90** is gated into U register where it will be subsequently executed as CI**90** during the next following machine cycle **52**.

In the event that the R_j content equals —1 during execution of an Index Jump instruction, then **Ā39–47** is disabled. Therefore, P register is in this case allowed to reference the permanent memory for NI**40** during machine cycle **51**. However, a 1 would still be added to the contents of R_j when it is rewritten back into the R memory so as to reduce its value to 0. Thus, unlike the Unconditional or Subroutine Jump instructions, the Index Jump instruction is conditional upon the presence of a number other than —1 in the referenced R memory location.

4.4. *Instruction Skip*

FIGURE 49 is a timing diagram showing the operation of the invention during execution of a Selective Skip instruction. As will be recalled, the "S" address portion of a Selective Skip instruction is held in U**10–23** where it is compared against a control panel indication to see if the skip condition is met. If the skip condition is met, then the next instruction sequentially following the skip instruction is not executed. It is assumed for purposes of illustration only that during machine cycle **35**, NI**21** acquired therein is a Select Skip instruction from the variable memory. It is also assumed that CI**20** being executed during machine cycle **25** has its "T" address portion referring to the permanent memory as the location of its operand. Therefore, overlap operation occurs during machine cycle **35** in that both NI**21** and CI**20** operand are acquired from two different memories.

By MP10 of machine cycle **35**, NI**22** is in U register with its Op Code 41 indicating that a Select Skip is to be performed. **Ā14–28** is thereby inhibited from transferring U6–9 to U*6–9 at MP12 time, inasmuch as No First R memory reference is required. At MP0 of machine cycle **36**, **Ā13–95** adds U**10–23** to R register (now empty) to place CI21 "S" address portion in U***10–23**.

At the beginning of machine cycle **36**, the P register contains the address of NI**22** which is assumed to be one in variable memory while P* register contains the variable memory address of NI**21**. Since a Select Skip operation is neither an operand or a jump instruction, **Ā39–58** gates P register into V register so that NI**22** will be acquired during machine cycle **36**. At this time in the machine cycle, it is not yet known whether or not a binary **1** is contained in the U**10–23** register stage selected by the control panel. Therefore, a memory reference is made of the next instruction at the address sequentially following the address of the Select Skip instruction, although later events might prevent the execution of this instruction during machine cycle **37**. Thus, the address in P register is incremented by **1** during MP times **3** through **6** in the normal manner such that at MP6 time, P register contains the address of NI23.

Assuming that the selective skip condition is met and that a significant negative output from Ā**10–61** is applied to N–**26**, the Selective Skip flip-flop **40–24** is set at MP4 time of machine cycle **36**. The setting of this flip-flop thereby indicates to the system that the execution of NI22 is to be skipped. Although NI22 will be placed in U register at MP10 time of machine cycle **36**, the now set Selective Skip flip-flop **40–24** enables Ā**40–27** to generate a signal at MP14 time. The output from Ā**40–27** in turn sets Skip FF**40–35** and enables Ā**40–21** (in the absence of an interrupt or a repeat operation) to clear U register stages **0–9** by Ō**40–19**. Therefore, both the operation code and "*j*" address portion of NI22 are cleared from U register by MP14 time of machine cycle **36**, which will prevent execution of NI22 during machine cycle **37**.

It is now possible to permit P register, during machine cycle **37**, to reference memory for NI23. This instruction will be acquired by U register at MP10 time of machine cycle **37** for execution by the system during machine cycle **38**. It will be noted that in machine cycle **36**, the clearing of U register stages **0–9** occurs after initiation of an index operation upon the "S" address portion of NI22, if this instruction calls for such. The results of this index operation are placed at MP0 time of machine cycle **37** into U***10–23** where it appears as the "T" address of CI22. Assuming that the "T" address portion refers to a variable memory location, and also assuming that the address of NI23 held in P register likewise refers to the variable memory, it is still possible to reference the variable memory by the P register inasmuch as Ā**39–58** generated the signal P To V in the absence of both a jump or an operand Op Code held in U**0–5**. Thus, the address of NI23 is gated to the V register at MP1 time of machine cycle **37** and V Instruction FF**39–14** is likewise set. There is no gating of the contents of U***10–23** at this time because there is no positive Operand signal applied to the appropriate gates in FIGURE 39. P register is incremented by 1 during MP3 through MP6 times of machine cycle **37** so that it now contains the address of NI24 which will be acquired, if possible, during machine cycle **38**. At MP10 time of machine cycle **37**, NI23 is gated from O register to the U register where it will be indexed, if required, and executed during machine cycle **38**.

FIGURE 50 is a timing diagram showing the operation of the invention if any one of the skip instructions 10, 11, 12, or 13 are being executed. The operation for any one of these four instructions is similar to that described above in connection with the Select Skip operation, except that the determination of the skip condition is performed by the arithmetic unit instead of in the U register. For example, the equality Skip instruction 10 requires the skip of the next instruction when the contents of the accumulator equals the contents of the memory location addressed by the "T" portion of the instruction. Likewise, the Non-Equal Skip instruction 11 requires the skip of the next instruction when the contents of the accumulator do not equal the contents of the "T" address location. Less Than Skip instruction 12 and Greater/Equal Skip instruction 13 respectively require the skip of the next instruction when the contents of the accumulator are less than or greater than or equal to the contents at "T." The arithmetic unit determines the condition for skip for these four instructions by subtracting the memory operand from the contents of the accumulator register after which the difference may be examined to determine its absolute value and sign.

In FIGURE 50, machine cycle **80** is the first considered where it is assumed that NI73 acquired therein is an Equal Skip instruction which is placed in U register at MP10 time. Currently being executed in machine cycle **80** is CI72, and overlap operation during machine cycle **80** is further assumed. Therefore, O register contains the CI72 operand. Since an Equal Skip operation is indexable, this is performed commencing at MP12 time of machine cycle **80** so that the "S" address portion of NI73 is transformed into a "T" address portion of CI73 which is subsequently placed at MP0 time into U***10–23**.

It is now assumed that the "T" address portion of NI73 refers to a permanent memory address location, while the contents of P register at MP0 of machine cycle **81** refers also to a permanent memory address location for NI74. Since any of the skip instructions now under consideration requires an operand, it is seen that in machine cycle **81** the permanent memory cannot be referenced simultaneously by both U***10–3** and the P register. Consequently, the overlap mode is inhibited during machine cycle **81** by preventinng the transfer of P register to the W register at MP1 time. Thus, both Ā gates **39–55** and **39–57** are inhibited from generating the signal P because of the presence of a permanent memory address in U***10–23** and the presence to an Operand Op Code in U**0–5**. Ā**39–59** does, however, gate U***10–3** to W register at MP1 time of machine cycle **81**. Ā**39–18** is also enabled to set W Instruction FF**39–13** which is subsequently cleared at MP2 time by Ā**39–15** which is in turn enabled to set W Operand FF**39–10**. Thus, W Instruction FF**39–13** will be cleared by MP10 time of machine cycle **81** so as to prevent the contents of Z register from being gated into the U register. The contents of Z register during machine cycle **81** will contain the operand used in performing the Equal skip instruction CI73, and will be gated from Z register to the arithmetic unit by means not shown so that it may be subtracted from the contents of the accumulator. Ā**39–15** also causes Ā**39–30** to set the Normal Advance PAC FF**39–31** at MP2 time of machine cycle **81**, which in turn prevents the incrementing of the contents of P register during MP3 through MP6 times. Therefore, P register continues to hold the address of the NI74 during machine cycle **81**.

At MP9 of machine cycle **81**, the presence of a skip instruction Op Code in U***0–5**, coupled with the absence of repeat mode operation, causes Ā**7–114** to set the PAC ADD FF**7–112**. This is automatically done at this time for any of the four arithmetic skip instructions under consideration. At MP9 time, U register is cleared of NI73 and thereafter remains empty because no instruction has been acquired from memory during machine cycle **81**. By MP14 time of machine cycle **81**, the arithmetic operation of subtraction has been completed so that the difference may be examined to ascertain the presence or absence of equality between the original accumulator contents and the CI73 operand. Assuming now that there is equality, the ZERO TEST Enable signal to Ā**40–31** is positive, which, when coupled with the positive Equal Skip signal from the command translator circuits of U***0–5**, enables an output from Ā**40–31** to be transmitted via Ō**40–34** to set Skip flip-flop **40–35**. The setting of Skip FF**40–35** therefore signifies that NI74 cannot be executed. However, due to the inhibition of overlap occurring during machine cycle **81**, it is seen that the contents of P register at the beginning of machine cycle **82** would normally be the address of NI74 which is to be skipped. Therefore, the signal from Ō**40–34** at MP14 time of machine cycle **81** also enables Ā**40–37** to generate a Clear P signal due to the fact that a negative signal Normal Advance PAC is applied thereto via Ō**40–40** because of the set condition of FF**39–31**. As may be seen from FIGURE 50, P is therefore cleared at MP14 time. The setting of Skip flip-flop **40–35**, coupled with an output from Ō**40–40**, also enables a signal to be generated from Ā**40–38** at MP15 time which transfers the contents of P* register to P register via the address counter. At this time it will be observed that P* holds the address of NI73. Furthermore, the setting of the PAC Add FF**7–112** at MP9 time allows the P* value to be incremented by 2 rather than 1. Thus, the value now residing in P register at the end of machine cycle **81** is the address of NI**75** rather than the address of NI**74**.

In machine cycle **82** P register references one of the memories, assumed here to be the permanent memory, to acquire NI**75** and place same in U register by MP10 time of this cycle. Since the Normal Advance PAC FF**39–31** is unconditionally cleared at the beginning of each machine cycle, the NI**75** address in P register can be incremented by 1 during the MP3 through MP6 times of machine cycle **82**. Therefore, P register will contain the address of NI**76** by the beginning of machine cycle **83**.

If it is instead assumed that the CI**73** operand does not equal the accumulator number, then $\overline{\text{A}}$**40–31** will not generate a signal at MP14 time. In this case, Skip FF**40–35** remains clear and signals Clear P and **P***$+n$ To P will not be generated from $\overline{\text{A}}$**40–37** and **40–38**, respectively. Therefore, the address of NI**74** in P register will not be changed at MP14 time so that it will be available at the beginning of machine cycle **82** to acquire NI**74** from memory and place same in the **U** register for execution during the machine cycle **83**. On the other hand, the situation may arise where overlap operation during machine cycle **81** is not inhibited because the CI**73** operand is acquired from a memory different from that holding NI**74**. In this case, operation during machine cycle **81** in FIGURE 50 would be quite similar to the operation in machine cycle **36** of FIGURE 49. That is to say, if the P register in FIGURE 50 contains an address for NI**74** in the variable memory, whereas the "T" address portion of CI**73** refers to the permanent memory, NI**74** would be acquired during machine cycle **81** and placed in U register at MP10 time. Since, in this case, the Normal Advance PAC FF**39–31** would not be set, P register will be incremented by 1 during machine cycle **81** so that it contains the address of NI**75** at the beginning of machine cycle **82**. Then, if the skip condition is met by MP14 time of machine cycle **81**, only $\overline{\text{A}}$**40–21** is enabled to clear **U0–9**. The absence of a significant positive signal from $\overline{\text{O}}$**40–40** (due to the clear condition of flip-flop **39–31**) prevents $\overline{\text{A}}$**40–37** and $\overline{\text{A}}$**40–38** from affecting the contents of P register at times MP14 and MP15 of machine cycle **81**.

As has been made apparent, the execution of one of the skip instructions 10, 11, 12, or 13 is quite similar to the execution of the Selective Skip instruction **41**, except that the latter never requires an operand from memory. Thus, it is normally possible during execution of a Select Skip instruction, according to FIGURE 49, to reference memory for a new instruction which is placed in U register during MP10 time. Usually, the incrementing by 1 of the address in the P register always occurs whenever P register is allowed to reference a memory for an instruction. This operation is shown in machine cycle **36** of FIGURE 49. However, there may be times during execution of the Select Skip instruction that the Normal Advance PAC FF**39–31** is set by conditions other than have so far been described, such as an interrupt grant or the like. In such a situation, the **P** register would not be incremented by 1, but instead would retain the address of the instruction sequentially following the Select Skip instruction, with P* register consequently retaining the address of the Select Skip instruction itself. If the Selective Skip FF**40–24** is set at MP4 time because of a skip condition, it will be noted that the negative output from the clear terminal of FF**40–24** is also applied via $\overline{\text{O}}$**7–115** to $\overline{\text{A}}$**7–114** in order to set the PAC ADD flip-flop **7–112** at MP9 time. This operation is indicated in machine cycle **36** of FIGURE 49 since the skip condition was assumed to be met. Now, if for some reason the Normal Advance PAC FF**39–31** is also set during machine cycle **36**, then the output from $\overline{\text{A}}$**40–27** at MP14 time enables signals from $\overline{\text{A}}$**40–37** and **40–38** at MP14 and MP15 times. Therefore, the contents of P* register would be incremented by 2 so that the **P** register at the beginning of machine cycle **37** contains the address of the instruction following the instruction to be skipped.

4.5. *Data Transfer*

FIGURE 51 discloses the timing diagram for the present invention when an input-output data transfer operation is requested to exchange information between the computer and its peripheral equipment. In general, upon acquisition from the computer memory of either an Initiate **F1** (31) or an Initiate **F2** (32) instruction, the "T" address portion of the instruction is transferred into either the **F1** or **F2** register where it is utilized to control the direction of transfer via the associated **C1** or **C2** register, as well as the particular peripheral unit involved. An indication that **C1** or **C2** can receive from or transmit a word to the variable memory forces an Input-Output Data Transfer instruction into the instruction register which acts to interrupt the main program for one machine cycle so that the variable memory can be referenced according to an address held in R memory. More than one data word transfer may be accomplished under the direction of the same Initiate **F1** or Initiate **F2** instruction, since the forcing of an Input-Output Data Transfer instruction into the instruction register also references the R memory at a particular address location therein which holds the address in the variable memory from which the data word is to be acquired or into which it is to be written. This V memory address held in the R memory is incremented each time a data word transfer is made, until it equals a number contained in another specific R memory location. After this, the transfer of data words under control of the previously withdrawn Initiate **F1** or Initiate **F2** instruction is terminated. Thereafter, any further data transfer requires that a new Initiate **F1** or Initiate **F2** instruction be acquired from memory during a normal instruction reference cycle and placed into the **F1** or **F2** register.

For purposes of this discussion, the first machine cycle to be considered in FIGURE 51 is **65**. During this time the address portion of NI**56**, which is assumed to an Initiate **F2** (32) instruction, is transferred from the instruction register to the **F2** register. NI**56** was withdrawn from memory during machine cycle **64**, and is thus assumed to be present in the U register by MP0 time of machine cycle **65** as shown in FIGURE 51. Since an Initiate **F2** instruction is indexable, the "S" address portion of NI**56** is added to the contents of the NI**56** "J" address location of R memory to place into U***10–23** the "T" address portion of CI**56**. It is also further assumed that an overlap operation occurred during machine cycle **64** so that P register now contains the address of NI**57**, assumed here to be in the permanent memory. P* register at the commencement of machine cycle **65** therefore still retains the address of NI**56**.

At MP0 time, the presence of an Initiate F2 operation code in **U0–5** causes $\overline{\text{A}}$**9–16** to generate a signal clearing various units of the Channel **2** registers and Control units such as may be found in FIGURES 31 through 38. Since an Initiate F2 instruction is not considered to be one requiring an operand from one of the memories, P register is thereby enabled to reference the permanent memory for NI**57** no matter what the value of the "T" address portion of CI**56**. Therefore, at MP1 of machine cycle **65**, both $\overline{\text{A}}$**39–55** and $\overline{\text{A}}$**39–57** generate a **P** To **W** signal, either one of which gates the contents of P register to the W register where said contents address the permanent memory for withdrawal of NI**57**. Also, $\overline{\text{A}}$**39–18** sets the W Instruction FF**39–13** so as to later enable $\overline{\text{A}}$**39–25** to gate the contents of Z into **U0–23**. Also at MP1 time of machine cycle **65**, $\overline{\text{A}}$**9–15** is enabled to gate the contents of **U*10–22** into the **F2** register, such that the **F2** register now contains the "T" address portion of CI**56** which is a thirteen bit number having a significance in **F2** previously described in connection with FIGURE 31.

Since P register references a permanent memory for NI57, at MP3 through MP6 times of machine cycle **65** its contents are transferred to P* register and then returned via the address counter so as to calculate the address of NI58 in the manner heretofore described. This function is, of course, performed by the successive enabling of Ā gates **39–36** through **39–39**. At MP9, **Ā41–24** clears U register of NI56 in preparation for receipt of NI57 from the Z register. At the same time, the presence of the Initiate F2 operation code now held in U***0–5** (transferred there from **U0–5** at MP4 time) causes **Ā31–38** to generate a signal setting flip-flop **F2–14** Go. The signal from **Ā31–38** is also applied via **N31–39** to **Ā31–40**, which in turn has positive inputs applied thereto from the clear and set output terminals of **F2–00** and **F2–01** respectively, if it is now assumed that these two stages indicate that the data transfer is to be made from the computer variable memory to the selected output peripheral unit. Therefore **Ā31–40** is also enabled at MP9 time of machine cycle **65** to generate the signal F2-Prime Output. This signal is applied to set the External Resume **FF33–10**, while the positive output from the now set flip-flop **F2–14** is applied to enable **Ā31–41** to generate the F2 Output And Go signal via **N31–42**. This latter signal remains positive as long as **F2–14** Go flip-flop is set.

At MP10 time of machine cycle **65**, NI57 is transferred from Z register into U register where its "S" address portion is indexed commencing at MP12 time upon transfer of the "J" address portion **U*6–9**. Thereafter, the R memory will be referenced to withdraw an index number for addition to the "S" address portion of NI57, thus resulting in the "T" address portion of CI57 placed in **U*10–23** register at the commencement of machine cycle **66**.

Due to the set condition of **FF33–10** and the presence of the F2 Output And Go signal from **Ā31–41**, **Ā33–11** is enabled at MP15 time of machine cycle **65** to generate a signal therefrom. This significant negative output from **Ā33–11** sets the External Resume lockout **FF33–12** as well as clearing the C Status **FF33–21** if it had previously been in a set condition. Furthermore, a signal from **Ā33–11** is applied via **Ō33–13** to clear the C2 register. Since the Terminate **FF33–34** is at this time cleared, **Ā33–11** also enables **Ā33–15** to generate a signal Request In/Out Data Transfer Channel **2** which is applied to **FF42–10**. Therefore, at the next following MP0 time, which is that occurring at the beginning of machine cycle **66**, **Ā42–22** is enabled to set the Priority 1 **FF42–16** which causes an interruption of the main program in order to perform a data word transfer from the variable memory to C2. From there the data word will be extracted and sent to the peripheral unit as selected by the value held in F2. Upon the setting of **FF42–16**, **Ā42–30** resets **FF42–10** at MP3 time as indicated in FIGURE 51. In addition, the now set condition of the External Resume Lockout **FF33–12** causes **FF33–10** to be cleared at MP1 time.

As may be seen from FIGURE 51, the instruction register now contains NI57 which may be executed during machine cycle **66**, while P register holds the address of NI58 which normally would be acquired during machine cycle **66** if overlap is possible. Assuming that the "T" address portion of CI57 calls for a reference of the variable memory, while the address of NI58 is in permanent memory, it is therefore seen that at MP1 time **Ā39–55** transfers P register to the W register, while **Ā39–60** transfers the contents of **U*10–23** to the V register. Also, **Ā39–18** sets W Instruction **FF39–13** which normally allows the transfer of NI58 from the Z register into the U register at MP10 time of machine cycle **66**. However, the now set condition of Priority 1 **FF42–16** generates significant signals from both **Ō42–37** and **Ō42–39** such that **Ā39–23** is enabled at MP3 time to clear the W Instruction **FF39–13**. Therefore, although the permanent memory is referenced during machine cycle **66** so as to withdraw NI58 and place same in the Z register, there will be no transfer of NI58 into the U register at MP10 time. At MP2 time, however, the V Operand **FF39–11** is set such that the CI57 operand in O register may be utilized in execution of CI57.

At MP5 of machine cycle **66**, the cleared condition of **FF33–10**, coupled with the absence of a positive signal C2 Resume, causes **Ā33–26** to clear **FF33–12**. At MP9, **Ā41–24** clears the U register of NI57 as usual. At MP10, an Input-Output Data Transfer Op Code 45, together with a "J" address is forced into **U0–9**. This function is performed by FIGURE 43 which is responsive to the positive signals Data Transfer and Data Transfer Interrupt from FIGURE 42 to enable Ā gates **43–10**, **43–11**, and **43–15** such that U register stages **0**, **3**, **5**, **6**, and **8** are set. **U10–23** remains empty at this time. At MP12 **Ā14–29** transfers **U6–9** to U***6–9** so that the latter now contains the "J" address of 10. This location in the R memory contains a 14 bit variable memory address of the first data word to be transferred from the variable memory to the C2 register. At MP13, RJ To R **FF16–36** is set to transfer this 14 bit number into the R register.

The presence of the positive Data Transfer/Interrupt signal from FIGURE 42 also enables **Ā39–32** to set the Normal Advance PAC Lockout **FF39–31** at MP2 time of machine cycle **66**. The setting of this flip-flop thereupon disables the sequential operation of Ā gates **39–36** through **39–39** so that the contents of P* and P register will not be changed during this cycle. Therefore, at the commencement of machine cycle **67**, P register holds the address of NI58 which is assumed, for the purposes of this discussion, to be one in the permanent memory. During machine cycle **67**, the variable memory is referenced at a location indicated by the number held in R register, so that it will be possible to reference permanent memory for NI58.

At MP0 of machine cycle **67**, **Ā13–95** causes the addition of **U10–23** (empty) to the contents of R register, with the sum begin placed in **U*10–23**. Therefore, the content of **U*10–23** at the commencement of machine cycle **67** is the same number as that in R register, which refers to a location in the variable memory holding the first data word to be transferred to the output unit. At MP1 time, **Ā39–60** transfers U***10–23** to the V register in the same way that an operand reference is made in the normal operation of the system. **Ā39–55** is also enabled to transfer the contents of P register to the W register, while **Ā30–18** sets the W Instruction **FF39–13**. At MP2 time, **Ā39–16** sets the V Operand **FF39–11** in the normal fashion. Therefore, NI58 is read from the permanent memory and placed into the U register, while the first data word to be transferred is read from variable memory and placed into O register. Also, since Priority 1 **FF42–16** is unconditionally cleared at MP12 time of machine cycle **66**, and since there is no inhibition of overlap during machine cycle **67**, Normal PAC **FF39–31** remains cleared to enable the address in P register to be incremented by 1 to the address of NI59. This latter function is accomplished by the successive generation of signals at MP3–MP6 times by respective Ā gates **39–36** through **39–39**.

At MP4 of machine cycle **67**, the input output code 45 now held in **U0–5** is transferred unconditionally to U***0–5** as shown in FIGURE 51. It will be remembered that U*6–9 already contains a decimal address of 10 indicated by the set condition of **U6** and **U8**. Since the operation code 45 is still held in **U0–5** at this time, a negative signal is generated by **N9–11** which prevents the clearing of U*6–9 by **Ā14–40** at MP4 of machine cycle **67**. Instead, **Ā14–34** is enabled at MP4 to set U* which changes U*6–9 contents from decimal 10 to decimal 11. R register is unconditionally cleared at MP4 time, however, in order to erase the contents previously extracted from

R memory location 10. Since the **RJ** To **R FF16–36** is unconditionally set at MP5 time, R memory is now referenced at location 11 to withdraw a number therefrom and place same into the R register. This number from R memory location 11 specifies a variable memory address location which is that holding the last data word to be transferred under control of the present contents of F2 register. If the contents of R Memory location 10 now held in U***10–23**, are equal to the contents of R Memory location 11, now held in the R register, this indicates that all of the data words have been transferred and that the input-output operation may cease. During MP15 of machine cycle **66**, **Ā9–10** is enabled to set the R Memory Add **FF17–52** so that the original content of **R10** is incremented by 1 as it is written back into R memory during MP1 through MP3 time of the next following machine cycle **67**. Therefore, each time that the contents of **R10** are acquired from R memory, a different number increased by 1 is placed back into **R10** so that eventually equality will be detected between the contents of **R10** and **R11**. This comparison is done by circuits of FIGURE 20 in a manner heretofore described. However, it is now assumed that equality is not present in machine cycle **67** between the contents of **R10** and **R11** so that further data word transfers must be performed under control of the **F2** register. Therefore, as soon as the external unit empties register C2 of the first data word, a request will be made to initiate another input-output data transfer operation which, when granted, will thereupon reference the variable memory at the address held in R10 to acquire the second data word to be transferred. The contents of **R10** will thereafter be incremented by 1 and replaced in **R10** with this procedure continuing until equality is detected between **R10** and **R11**.

At MP9 time of machine cycle **67**, **Ā41–24** clears the U register of the input output code 45 and RJ address of 10 in preparation for a transfer of NI58 from Z register thereto. This occurs at MP10 time as shown in FIGURE 51. Furthermore, data word 1, now residing in O register, is transferred to C2 register by the following circuitry. The set condition of U***8** indicates that a channel **2** transfer operation is begin performed. The positive signal thus generated from the set output terminal of U***8**, coupled with the positive signals F2-Output and Instruction 45, enable **Ā12–26** to emit a significant negative output which, when inverted, enables **Ā12–25** to generate a signal Channel 2 Ready at MP9 time of machine cycle **67**. This signal in turn is applied to **Ā** gates **32–10** to **32–34** to gate the contents of O register into C2 register from whence they are available to be transmitted to the output unit. In addition, the Channel **2** Ready signal from FIGURE 12 is applied via **N33–20** to **Ā33–19**, having its other input signal positive because of the set condition of **F2–14**. Therefore, C Status **FF33–21** is set at MP9 time of machine cycle **67** which in turn enables **Ā33–22** to generate a **C2** Ready signal via **Ō33–24**. This C2 ready signal is transmitted to the selected external unit and indicates to it that C2 register now contains a data word which may be sampled by the output unit at its leisure. In the meantime, the main program continues in normal fashion such that **NI58** now held in the U register is indexed, if required, and executed during machine cycle **68**, with further instructions being acquired in sequence and executed until the output unit requests the second data word. This request for the second data word is indicated by either a positive signal **C2**-Resume (from a 24 bit capacity external unit) applied to **N33–25**, or by a negative signal DSC Output Resume if a 6 bit unit (tape or typewriter) is selected. In the case of the former, after inversion by **N33–25** and **N33–33**, a positive signal is applied to **Ā33–32**, which in turn switches with the positive signal **F2**–Go and a positive signal from the clear output terminal of **FF33–12**. Since the unit may generate a positive **C2** Resume signal at various times within a computer machine cycle a series of machine pulses 1, 5, 9, and 13 are provided in order to insure that the positive C2–Resume signal does not disappear before **Ā33–32** can be clocked. Therefore, **Ā33–32** will be enabled to generate a signal sometime during a machine cycle in which the C2–Resume is provided in order to set **FF33–10**. Setting of this flip-flop will enable **Ā33–11** to generate a signal at the next following MP15 time which in turn sets External Resume Lockout **33–12**, clears the C2 register via **Ō33–13**, and clears **FF33–21** which indicates now that C2 register is emptied of the first data word. In addition, the signal from **Ā33–11** enables **Ā33–15** to request another input-output data transfer on channel **2** which is applied to the priority circuits and causes the setting of Priority 1 **FF42–16** in the manner previously described in connection with machine cycle **66** of FIGURE 51. With the setting of Priority 1 flip-flop **42–16**, the main program is again interrupted in exactly the same way as shown in machine cycles **66** and **67**. The second data word is found in the variable memory address having a value **1** more than the address holding the first data word. An interruption of the next instruction acquisition will occur for one cycle to allow the forcing of the input-output code **45** into the instruction register and the subsequent transfer of the second data word from O register to C2 register. This operation is therefore repeated until the content of R Memory location 10 is equal to the content of R Memory location 11.

It is now assumed for purposes of this discussion that only two data words may be transferred under the control of the number placed in F2 register during machine cycle **65** of FIGURE 51. Therefore, the initial content of R memory location 10, as withdrawn during machine cycle **66**, is 1 less in value than the content of **R11**. Therefore, the transfer of the initial **R10** content through the R address counter in machine cycle **67** causes a number equal to the contents of **R11** to be written back into **R10** during MP1 through MP3 times of machine cycle **67**. Thereafter, upon request by the output unit for the second data word, the referencing of the R memory location 10 causes the second data word to be withdrawn from a variable memory address having the same value as that held in R memory location 11. Assuming now that the request for the second data word is made of the computing system during machine cycle **70**, it will be seen that during machine cycle **71** the input output code **45** is forced into the U register where it is also present during machine cycle **72** which is that in which the variable memory will be referenced for the second data word. Thus, FIGURE 51 shows a portion of machine cycle **72** beginning at MP4 time in which it is assumed that the input output code 45 is present in the U register, and that the content of R Memory 10 has been placed in U***10–23** where they were used at the beginning of machine cycle **72** to initiate reference of the variable memory for the second data word. At MP4 time of machine cycle **72**, U***6–9** contents are changed from 10 to 11 in the same manner as shown at MP4 time of machine cycle **67**. The R memory is now referenced at location 11 so that its contents are transferred to R register and compared with the contents of **R10** now residing in U***10–23**. Under the assumption given above, the numbers held in U***10–23** and in the R register are now equal so that a positive signal is generated from **N20–11**. This signal in turn is applied to **Ā12–31** having other inputs thereto the same as those applied to **Ā12–26**. Therefore, at MP8 time of machine cycle **72**, **Ā12–31** generates a significant signal Terminate Channel **2** which sets the Terminate flip-flop **33–34**. As in machine cycle **67**, **Ā12–26** also generates a continuous significant negative output so that **Ā12–25** generates a Channel **2** Ready signal at MP9 time of machine cycle **72**. This signal gates the second data word from the O register to C2 register and sets the C Status FF**33–21** to indicate that C2 register is again filled with information. The other operations during machine cycle **72** are exactly similar to those in machine cycle **67**. Subsequent to completion of machine cycle **72**, execution of the main program instructions continues as before until the output unit receives the second data word held in C2 and transmits a request to the computer system for a third data word.

Assume now that a positive C2–Resume signal is supplied during machine cycle **76** so that the External Resume FF**33–10** is again set by at least MP13 time. Thus, at MP15 time, Ā**33–11** sets FF**33–12**, clears C2 register, and clears C Status FF**32–21**. However, since the Terminate FF**33–34** is now set, no signal is generated by Ā**33–15** at this time. Therefore, FF**42–10** cannot be set in the priority unit. Instead, the output from Ā**33–11** at MP15 time of machine cycle **76** is gated through Ā**33–35** in order to set the Drop Go flip-flop in the F2 register. The setting of this flip-flop thereupon causes Ā**31–47** to clear the F**2–14** Go flip-flop at the next MP1 time in machine cycle **77**. At MP3 time of machine cycle **77**, Ā**31–51** is now enabled by the cleared condition of **F2–14** to clear **F2–13**. The clearing of **F2–14** thereby terminates the positive signals F2–Go, and F2 Output And Go. Terminate FF**3–31** remains set until another Clear inout Channel 2 signal is received from FIGURE 9, which occurs only when an Initiate F2 instruction is withdrawn from memory and placed into the U register in some future NI acquisition cycle.

In recapitulation, it may be said that the transfer of one or more data words under control of a number held in the F2 register is interspersed with the acquisition and execution of the main program instructions. Upon acquisition of an Initiate F1 instruction from memory, exactly the same procedure applies to transfer of information between the computer and C1 register as has been described in connection with channel **2**. Therefore, no detail description will be given of the data transfer in channel **1** since it should be obvious from the preceding descriptions.

4.6. *Interrupt*

FIG. 52 is a timing diagram showing the operation of the invention when the computer is requested to interrupt the main program due to the setting of one of the Request Interrupt flip-flops **42–12** to **42–15**. This operation is similar to that previously described in connection with the sub-routine jump, in that the main program is interrupted so that a sub-routine program can commence whose first instruction address is specified by "S" address portion of the Interrupt instruction. Also, the next instruction address held in **P** register is stored in the R memory at the location indicated by the "J" address portion of the Interrupt instruction, so that upon termination of the interrupt sub-routine, the system returns to the main program. An Interrupt instruction may be programmed, i.e., an instruction with Op Code 46 may be stored within one of the memories, but is normally generated by the main control as required. The latter technique is the one illustrated in FIGURE 32, which specifically shows the execution of an Interrupt instruction generated because of error detection by the E register. However, the execution of all other types of Interrupt instructions is similar to the execution of the Error Interrupt, except for the specific values utilized in the "J" and "S" address portions of the Interrupt instruction.

Referring first to FIGURE 42, the generation of an Error Interrupt signal from the E register causes Ā**44–19** to set **FF42–12** which requests the priority circuits for an interruption of the main program to perform an error sub-routine. The setting of **FF42–12** is here assumed to occur in some machine cycle prior to machine cycle **97** which is the first to be shown in FIGURE 52. The operation of the priority circuits, as regards lockouts of six machine cycles, has previously been described in connection with FIGURE 42. Assuming now that by MP0 time of machine cycle **97** an error interrupt priority may be granted, it is noted that Ā**42–24** sets the priority 3 **FF42–18** at MP1 time of machine cycle **97** in FIGURE 52.

It is further assumed in connection with machine cycle **97** that during machine cycle **96**, NI**56** had been withdrawn from memory and placed in the U register for execution during machine cycle **97** as CI**56**. Therefore, at commencement of machine cycle **97**, it is seen that **U0–23** holds NI**56** while U***6–9** holds the CI**56** "J" address. In addition, assuming that NI**56** is indexable, then R register holds the index number which is added to **U10–23** at MP0 time of machine cycle **97** in order to place into U***10–23** the CI**56** "T" address portion. It is assumed that this address refers to the permanent memory in which is stored the operand to be used during the execution of CI**56** in machine cycle **97**. It is also further assumed, for purposes of this description, that overlap occurred during the preceding machine cycle **96** so that the P register address was incremented from the NI**56** address to the NI**57** address, with the NI**56** address being maintained in P* register. Assuming that the NI**57** address contained the P register at MP0 time of machine cycle **97** refers to the variable memory, it is therefore seen that overlap is possible in machine cycle **97** because NI**57** and the CI**56** operand will be acquired from different memories.

Returning now to the detailed description of the interrupt operation, at MP1 time of machine cycle **97** Ā**39–20** sets the V Instruction flip-flop **39–14**, while Ā**39–56** transfers P register into V register such that the latter now contains the address for instruction NI57. Additionally, since the CI**56** operand is acquired from the permanent memory during machine cycle **97**, Ā**39–39** transfers U***10–23** into the W register, while at MP2 time Ā**39–15** sets the W Operand FF**39–10**. With regard to the address in the permanent memory, Z register is filled at MP8 time of machine cycle **97** with the CI**56** operand, which is thereupon made available to the arithmetic unit or other portion of the system for use during execution of CI**56**. However, although the V Instruction FF**39–14** is set at MP1 time of machine cycle **97**, it will be noted from FIGURE 52 that the setting of Priority 3 FF**42–18** above described enables the Ā gate **39–23** to clear V Instruction FF**39–14** at MP3 time of machine cycle **97**. This is so, since the set condition of **FF42–18** causes the generation of the significant positive signals Data Transfer/Interrupt from Ō**32–29** as well as from **N42–42**. In addition, the positive Data Transfer/Interrupt signal is applied to Ā**39–32** via Ō**39–33** to set the Normal Advance PAC Lockout FF**39–31** at MP2 time of machine cycle **97**. The setting of FF**39–31** thereby prevents the enabling of Ā gates **39–36** to **39–39** at MP times 3, 4, 5, and 6, respectively, so as to prevent change of the NI57 address in the P register. Therefore, P register continues to maintain the address of NI57 throughout machine cycle **97** and until the commencement of machine cycle **98**.

As mentioned above, the CI**56** "T" address portion in W register during machine cycle **97** causes the referencing of permanent memory at this location by WX Shaper flip-flop **27–10**, SW Shaper FF**27–11**, and permanent memory Strobe FF**27–12** in order to read into the Z register the CI**56** operand. V register also contains the address of NI**57** and the V Reference FF**23–10** is set at MP1 time of machine cycle **97** by the signal from Ā**39–56**. Therefore, the read and write flip-flops **43–13** and **23–24**, respectively, are able to reference the variable memory at this address location in order to place NI**57** into O register. As will be recalled from previous descriptions, the instruction residing in O register would normally be gated to U register at MP10 time of machine cycle **97** if the V Instruction FF39–14 were still set. However, the clearing of FF39–14 at MP3 time prevents generation of a gating signal O To U0–23 from Ā39–26.

Instead of NI57 being placed in U register at MP10 time of machine cycle **97**, an Error Interrupt instruction is forced therein having an Op Code value of 46, a "J" address portion of 15, and an "S" address portion of 10000 in accordance with the table previously described. The set condition of Priority 3 **FF42–18** at MP10 time of machine cycle **97** causes output signals to be generated from Ā gates **43–14**, **43–11**, **43–15**, **43–13**, and **43–16**, which respectively set the U register stages **0**, **3**, **4**, **6**, **8**, **9**, **7** and **11**. The setting of these U register stages at MP10 time of machine cycle **97** thereupon places the Error Interrupt instruction into the instruction register whose "S" address portion 10000 indicates the permanent memory address location of the first instruction to be performed during the Error Interrupt sub-routine, while the "J" address portion 15 indicates the R memory location in which is to be stored the address of NI57 from the P register. The latter function is performed so that at the end of the error interrupt sub-routine, an Unconditional Jump Instruction will cause the R memory to be referenced at this location to withdraw the address of NI57 therefrom for use in returning the system to the main program.

At MP12 time of machine cycle **97**, **Ā14–29** gates the "J" address 15 from **U6–9** to **U*6–9**, where it is now available to address the R memory so that the contents thereof may be withdrawn and placed into the R register at MP13 time. Thus, **Ā16–38** sets RJ To R flip-flop **16–36** at MP13 time which reads the contents of R memory location 15 into the R register. However, the RJ contens are not used to index the "S" address portion of the error instruction in **U10–23**. Instead, the positive signal Interrupt Instruction from FIGURE 9 is applied to **Ā16–35** so that the latter is enabled at MP15 time to clear R register of the contents previously placed therein at MP13. The positive Interrupt Instruction signal is also applied via **N16–27** to **Ō16–26** to enable **Ā16–23** to gate the contents of P register into R register at MP0 time of machine cycle **98**. Therefore, at the commencement of machine cycle **98**, R register contains the address of **NI57** which is thereafter stored back into R memory at location 15 between MP1 and MP3 times due to the setting of R To OR Gates flip-flop **18–93** by **Ā18–94**. This storing of the contents of P register into R memory is identical to that previously described in connection with the Unconditional Jump instruction.

As illustrated; by FIGURE 52, **U*10–23** now holds the memory address 10000 of the first instruction to be performed during the error sub-routine program. Likewise at MP0 time of machine cycle **98**, P register still retains the memory address of **NI57**, which is assumed for purposes of this discussion to be in the variable memory. However, the referencing of the variable memory by P register is inhibited. The presence of the interrupt Op Code Code 46 in **U0–5** at the beginning of machine cycle **98** causes **Ō9–24** to emit a positive signal Interrupt, SRJ, UCJ, which, when applied to **Ō39–48** via **N39–49** makes positive the Jump output line and further disables **Ā39–56** via **N39–50**. The negative output from **N39–50** is further applied to **Ō39–51** to make positive its output which in turn disables **Ā39–58** via **N39–53**. Therefore, no signal P To V may be generated at MP1 time of machine cycle **98**. Likewise, the absence of a positive signal to **Ā39–20** prevents this gate from setting the V Instruction **FF39–14**. However, due to the positive output from **Ō39–51** and the presence of a W memory address in **U*10–23**, **Ā39–59** generates U* To W to transfer the contents of **U*10–23** into the W register. The W register now holds the "S" address portion of the Error Interrupt instruction and may also be considered, for purposes of this discussion, as the address of **NI85**. Permanent memory is now referenced during machine cycle **98** and **NI85** acquired therefrom into the Z register from whence it will be transferred to U register during MP10 time of this cycle.

It is also necessary to change the contents of P and P* registers in order that they may sequentially withdraw the instructions comprising the error sub-routine. This operation commences at MP2 time of machine cycle **98** when **Ā5–31**, which is responsive to the presence of an output from **Ō39–48**, clears P register of the **NI57** address. **Ā5–25** thereupon gates **U*10–23** to P register at MP3 time so that the latter now contains the address of **NI85**. Because the normal advance PAC **FF39–31** is unconditionally cleared at the beginning of machine cycle **98**, Ā gates **39–36** through **39–39** may now increment the address contained in P register during MP times 3–6 in the normal manner. Thus, **Ā39–36** clears P* at MP3 while **Ā39–37** transfers P to P* at MP4 time. P* register now contains the address of **NI85**. At MP5, **Ā39–38** clears P register of the **NI85** address and **Ā39–39** transfers the contents P* to P register via the address counter which increments P* contents by **1**. **P** register now contains the address (10001) of **NI86**, which is the second instruction to be performed during the error interrupt routine. Therefore, during machine cycle **99**, an attempt will be made by P register to reference the permanent memory for **NI86** unless overlap is inhibited during this cycle.

At MP9 time of machine cycle **98**, **Ā41–24** clears **U0–23** so that **NI85** from Z register may be placed therein at MP10 time by **Ā39–25**. If this is an instruction whose "S" address portion requires indexing, such operation will commence at MP12 time when **U6–9** is transferred to **U*6–9** by **Ā14–29**. Execution of **NI85** is accomplished during machine cycle **99**.

Although their operation is not specifically shown or described in connection with FIGURE 52, it is to be understood that the granting of the Error Interrupt initiates operation of the lockout enable flip-flops **1** through **4** as previously described. Thus, at MP3 time of machine cycle **97**, **Ā42–43** sets **FF42–44** so that a request for any other interrupt operation cannot be granted for six machine cycles thereafter. During this time, the error interrupt sub-routine is initiated. However, the operation of lockout flip-flops **1** through **4** in FIGURE 42 does not prevent input-output data transfers from being made, since such can be interspersed between sub-routine instructions as well as between main program instructions.

4.7. *Repeated instruction*

FIGURE 53 shows the operation of the programming system for execution of a Repeat instruction **40** and the execution of the repeated instruction which is that following the Repeat instruction. In general, a Repeat instruction **40** causes the instruction following it to be executed a number of times specified by the contents of address location 7 in the R memory. Therefore, the instruction immediately following a Repeat instruction **40** in the programming sequence is referred to as the repeated instruction. If the repeated instruction requires indexing each time that it is executed, such that its "T" address portion is changed, then the increment of indexing is the "S" address portion of the repeated instruction while the number held in the R memory location specified by the "J" address portion of the repeated instruction contains the base reference address minus the increment. Furthermore, during execution of the repeated instruction, the address of the instruction following the repeated instruction is calculated and held in **P*** register, while the memory address from which the Repeat instruction **40** itself was acquired is recalculated and stored in P register. This latter function is performed so that, if an interrupt sub-routine is granted during the repeat operation, the address of the Repeat instruction 40 can be placed in R memory for use at termination of the Interrupt sub-routine in completing the required number of executions of the repeated instruction.

For purposes of this discussion, the first machine cycle **73** considered in FIGURE 53 is that during which the Repeat 40 instruction is acquired from memory as **NI41**. Assuming that overlap was performed during machine cycle **73**, **CI40** is currently being executed so that U***0–5** contains the **CI40** Op Code, while U***10–23** holds the "T" address portion of **CI40**. It is also assumed that the **CI40** operand is found in the variable memory whereas **NI41** is acquired from permanent memory. In this case, P register has been incremented during machine cycle **73** so that by MP6 time it contains the address of **NI42**, the repeated instruction.

At MP9 time of machine cycle **73**, **Ā41–24** clears **U0–23** so that **Ā39–25** may gate the contents of Z into **U0–23** at MP10 time. **U0–5** now contains the operation code of **NI41** which is assumed to be octal 40, that of the Repeat instruction. Since Repeat instruction 40 is not indexable, the "J" address portion of **N41** may or may not contain a number other than 0. Furthermore, the "S" address portion of **NI41** may or may not contain a number other than 0 inasmuch as it is not utilized during execution of the Repeat 40 instruction during machine cycle **74**. At MP12, R register is cleared unconditionally. However, the presence of a Repeat 40 operation code in **U0–5** causes the generation of a positive signal No First R Memory Reference from **Ō9–30** to inhibit operation of **Ā14–29** at MP12 time. Thus, there is no transfer of the contents of **U6–9** to **U*6–9** such as normally occurs. Instead, **Ā14–39** insures that **U*6–9** is cleared at MP12 time of machine cycle **73**. Furthermore, Op Code 40 in **U0–5** inhibits operation of **Ā16–38** such that RJ To R flip-flop **16–36** remains cleared at MP13 time of machine cycle **73**. There is therefore no reference of the R memory during the latter portion of machine cycle **73**.

At MP13 time, **Ā10–62** is responsive to the Repeat instruction in U register to set all of the U register stages **10–23**. In the binary complement arithmetic used in the present computing system, this is equivalent to setting the value −1 into these U register stages. Thus, it makes no difference what the value is of the "S" address portion of **NI41**, inasmuch as all of the **U10–23** stages will be set at MP13 time. In addition, **Ā16–29** is responsive to the Repeat 40 code in U register to gate the contents of P register to R register, such that the latter now holds the address of **NI42**. At MP0 of machine cycle **74**, **Ā13–95** causes the **U10–23** contents to be added to R register and the sum placed into **U*10–23**. This effectively results in 1 being subtracted from the address of **NI42** so that the number placed into **U*10–23** is the address of **NI41**, the Repeat instruction itself. This is so because **NI41** and **NI42** are held in sequential address locations.

Although **U*10–23** and P register now contain addresses referring to the permanent memory, a Repeat 40 Op Code in **U0–5** is not considered to require an operand from memory, so that P register will not be inhibited from referencing permanent memory for **NI42**. Therefore, at MP1 time of machine cycle **74**, **Ā49–57** gates the contents of P register to W register for referencing of the permanent memory. There is, however, no gating of **U*10–23** to the W register at this time.

NI42 is subsequently read from the permanent memory and placed into Z register, from whence it is transferred at MP10 time of machine cycle **74** to the U register. This transfer is possible because the W Instruction **EE39–13** is set at MP1 time by **Ā39–18**, so that **Ā39–25** is enabled at MP10 time.

Ā39–32 is responsive to the presence of a Repeat Op code 40 in U register to set the Normal Advance PAC **FF39–31** at MP2 of machine cycle **74**. In addition, **Ā7–113** is likewise enabled at this time to set the PAC ADD **FF7–112**, which increments by 2 any address transferred from P* to P register via the address counter. **Ā7–13** also sets the Extend Repeat **FF7–116**. The setting of Normal Advance PAC **FF39–31** disables the sequential outputs from Ā gates **39–36** through **39–39** such that the P address will not be changed in the normal fashion. Instead, **Ā5–32** is responsive to the Repeat 40 Op Code in **U0–5** to clear P register at MP3 time. At MP4 of machine cycle **74**, **Ā7–104** thereupon gates the contents of P* register through the address counter to P, whereupon the **NI41** address of P* register is incremented by 2 so that P register now holds the address of **NI43**. At MP6 time, **Ā6–28** clears P* register because of the Repeat 40 Op Code, and **Ā6–25** transfers P register to P* register at MP7 time. P* register now holds the ddress of **NI43**, which is the instruction to be executed following the repeated execution of **NI42**. At MP8 of machine cycle **74**, **Ā5–30** clears P register of **NI43** address. The enabling of **Ā5–26** at MP9 time transfers the contents of **U*10–23** into P register. P register now contains the address of **NI41**, which is the Repeat 40 instruction previously acquired during machine cycle **73**. If an interrupt is granted during the repeated execution of **NI42**, P register is gated into R memory so that upon termination of the interrupt sub-routine, the **NI41** address may be acquired therefrom and used to acquire **NI41** again so that the required number of repeated executions of **NI42** may be accomplished.

At MP3 time of machine cycle **74**, **Ā41–11** responds to the Repeat 40 Op Code in U register to set the Repeat mode **FF41–10** which remains set during the repeated execution of **NI42**. R register is unconditionally cleared of the **NI42** address at MP4 time, while **Ā14–33** is responsive to Repeat mode **FF41–10** to force a decimal 7 into **U*6–9** by setting U* register stages **7**, **8**, and **9**. Address location 7 of R memory contains a number specifying the number of times that **NI42** is to be repeated. This number in R memory is expressed in complementary fashion such that the addition of 1 thereto during each execution of **NI42** results in it being eventually diminished to a value of −1. This computation is performed by withdrawing the contents of R memory location 7 during MP5 time and transferring same to R register. At MP7, **Ā41–14** sets the R Memory ADD **FF17–52** so that upon writing the contents of R register back into R memory at MP9 time, 1 will be added thereto. Thus, for each execution of the repeated instruction **NI42**, the content of R memory location 7 is withdrawn, incremented by 1, and replaced in R memory location 7, until eventually this content equals −1. This signifies that **NI42** will have been executed the required number of times by the end of the next following machine cycle so that **NI43** can thereafter be acquired and executed.

It is assumed for this discussion that the first reference made of R memory location 7 during machine cycle **74** finds that its contents initially do not equal −1. This is indicated in FIGURE 53 by showing that the number held in R register commencing MP5 time of machine cycle **74** does not equal −1. The content of R register is transferred back to R memory location 7 by setting of R To OR gates **FF18–93** by **Ā18–95** between MP9 and MP12 times of machine cycle **74**. Also at MP9 time, **Ā41–24** clears U register and **Ā39–25** gates the contents of Z into **U0–23** at MP10 time, whereupon the latter now holds **NI42**. **NI42**, which is the repeated instruction, will be executed once as **CI42** during machine cycle **75**. If this instruction is indexable, then **U6–9** is gated to **U*6–9** at MP12 time by **Ā14–29**. At MP13 time of machine cycle **74**, the setting of RJ To R **FF16–36** references R memory at the location specified by the "J" address portion of **CI42** now residing in **U*6–9**. The number from R memory is transferred to R register where it is added to the contents of **U10–23** at MP0 time of the next following machine cycle 75. However, as before mentioned, the "S" address portion of NI42 residing in U10–23 is a constant increment value which is added to the RJ number each time that NI42 is executed. It is therefore seen that the RJ number changes for each NI42 execution by an increment equal to the "S" address portion of NI42. For example, if the initial value of the RJ location is 04320 (a variable memory address) and the "S" address portion of NI42 is 00001, then the operand required for the successive executions of NI42 will be withdrawn from variable memory address locations beginning at 04321, followed by 04322, 04323, etc. Thus, referring again to FIGURE 53, Ā13–95 adds R to U10–23 and places the result in U*10–23, with the latter now holding the "T" address portion of CI42 to be used as reference memory for the operand required for the first execution of NI42. However, instead of R register being read back into R memory location RJ during machine cycle 75, Ā18–105 sets U*10–23 To Or Gates FF18–104 at MP1 time to gate the contents of U*10–23 into R memory at the location specified by the "J" address held in U*6–9. Therefore, the original content of RJ, as modified by the addition of the increment S, is placed back into R memory. If a second execution of NI42 is required, the modified content of RJ will again be withdrawn and incremented by the "S" address portion, with this new value being replaced in R memory for use in further executions of NI42.

At MP12 time of machine cycle 74, Ā41–13 sets the Next Instruction Lockout FF41–12 because of the Repeat 40 Op Code contained in U*0–5. The setting of FF41–12 prevents clearing of U register at MP9 time by Ā41–24. Therefore, the repeated instruction NI42 remains in U0–23 stages as long as is required for the repeated executions, unless there is an interrupt or a data transfer request made while the repeated executions are being performed. At MP1 time of machine cycle 74, the "T" address portion of CI42 references the variable memory for the first operand required in the first execution of CI42, but P is prevented from referencing the permanent memory for the NI41 address currently held by it. The first operand is thereby acquired from the variable memory and gated into O register for use by the system in the execution of CI42.

During MP4 of machine cycle 75, R register is again unconditionally cleared and a decimal 7 forced into U*6–9 by Ā14–33. R memory address location 7 is thereby referenced and its number extracted and placed into R register to determine if its value is equal to −1. If so, then NI42 will be executed just one more time during machine cycle 76 before the repeat operation is terminated. However, it is assumed in the present discussion that NI42 must be executed three times. Therefore, the content of R memory location 7 during machine cycle 75 does not equal −1, as indicated in FIGURE 53. At MP7 time of machine cycle 75, Ā41–14 again sets R Memory Add FF17–52 so that the content of R register is incremented by 1 upon transfer back to the R memory at MP9 through MP11 time of the cycle.

At MP12 time of machine cycle 75, the contents of U6–9 are double gated to U*6–9 such that the latter now contains again the "J" address portion of CI42. R register is unconditionally cleared at MP12, and the R memory referenced to withdraw the "T" address portion of CI42. At MP0 time of machine cycle 76, the "S" address portion of U10–23 again increments the value contained in R register with the result being placed in U*10–23 as the "T" address portion of CI42. U*10–23 now contains, by MP0 time of machine cycle 76, a value differing from the value it held in machine cycle 75 by an increment equal to the "S" address portion of NI42. Thus, during machine cycle 76, the variable memory will be referenced at the address held in U*10–23 to acquire the operand necessary for the second execution of CI42. Since the Next Instruction Lockout FF41–12 continues to remain set during machine cycle 76, P register is again prevented from referencing the permanent memory. In addition, the continued setting of Repeat Mode FF41–10 sets Normal Advance PAC FF39–31 at MP2 to prevent any change in the contents of P* and P registers. U*10–23 is thereupon transferred to V register at MP1 time of machine cycle 76, and at the same time is transferred to R memory location RJ by the setting of FF18–104.

Under the assumption that CI42 is to be executed three times in succession, the addition of 1 to the content of R register during machine cycle 75 results in a value of −1 being written back into R memory location 7 by MP11 time of this cycle. Therefore, when R memory location 7 is referenced during machine cycle 76, the number placed into R register is equal to −1. However, CI42 is executed for only the second time during machine cycle 76. Therefore, CI42 must be executed once more during the next following machine cycle 77, after which the repeat operation is terminated. The detection of a −1 value in R register during machine cycle 76 thus initiates functions which provide termination during the next machine cycle 77. Ā41–17 is responsive to the R register value −1 to set Terminate FF41–16 at MP8 time of machine cycle 76. Thereafter, Ā41–18 transfers P* to P at MP13 time, so that the latter register now holds the address of NI43 which is the instruction sequentially following the repeated instruction NI42. At the same MP13 time, Ā41–20 clears the Next Instruction Lockout flip-flop 41–12 so that P register can reference memory during machine cycle 77 is overlap is possible at that time. During the latter part of machine cycle 76, NI42 is indexed, as was done in machine cycles 74 and 75, in preparation for the third and final execution of CI42 during machine cycle 77.

At the commencement of machine cycle 77, U*10–23 contains the modified "T" address portion of CI42 which is here assumed to reference the variable memory. P register, on the contrary, now contains the address of NI43 which is here assumed to reference the permanent memory. In addition, the Next Instruction Lockout FF41–12 is cleared by this time so that a P=W signal is generated from the address range translator in FIGURE 5. Since both U* register and P register are referencing different memories during machine cycle 77, overlap is possible to acquire NI43 for transfer into U register by MP10 time of machine cycle 77. Also, Ā41–22 is responsive to the setting of Terminate FF41–16 to clear Repeat Mode FF41–10 at MP0 time of machine cycle 77. FF41–16 in turn is cleared by MP3 time of this cycle. The clearing of Repeat Mode flip-flop 41–10 thereupon prevents U*6–9 from referencing R memory location 7 as was done during machine cycles 74, 75, and 76. Consequently, the programming unit of the invention operates in normal fashion during machine cycle 77 when final execution of the repeated instruction is performed, in that the next instruction is acquired if two different memories are referenced. At MP9 time of machine cycle 77, Ā41–24 is again enabled to clear U register of NI42 in preparation for the gating of NI43 from Z register to the U register at MP10 time. In addition, the clearing of the Repeat Mode FF41–10 prevents the setting of the Normal Advance PAC FF39–31 during machine cycle 77, so that P register has its address incremented to that of NI44.

FIGURE 54 is a timing diagram showing a modification of the repeat operation if the repeated instruction is one of the skips 10, 11, 12, or 13. As will be recalled from the description of the Skip instruction, a skip of the next instruction following the Skip instruction is made if the skip condition is met. When a Skip instruction 10, 11, 12, or 13 follows a Repeat instruction 40, then the skip instruction is executed repeatedly and normally uses a different operand for each execution because of indexing. The repeat mode is terminated when the skip condition is sensed or the number contained in R memory location 7 equals —1 whichever occurs first. In the event that a skip condition is met during one of the execution cycles, the repeat operation is terminated and the program skips the next following instruction. On the other hand, if the repeated Skip instruction is executed the full number of times without the skip condition being found, the repeat operation is terminated in the normal fashion and the instruction following the skip instruction is acquired and executed.

In FIGURE 54, the leftmost timing diagram, let it be assumed that **NI42** is an Equal Skip instruction 10 which has been acquired in the preceding machine cycle **74** in the manner shown in FIGURE 53. As described above in connection with machine cycle **75** of FIGURE 53, **NI42** is executed for the first time during this cycle wherein it is also determined that the content of R memory location 7 is not equal to —1. However, in FIGURE 54 now assume that during the first execution of **NI42** in machine cycle **75**, the skip condition of equality is found. That is, the **CI42** operand placed in the O register during machine cycle **75** is equal to the content of the accumulator. The sensing of the skip condition is performed at MP14 time when **Ā40–31** sets Skip **FF40–35**. Since both the Repeat Mode **FF41–10** and the Normal Advance PAC **FF39–31** are set in machine cycle **75**, **Ā40–37** is enabled to clear P register of the **NI41** address at MP14 time. Thereafter, **Ā40–38** transfers the contents of P* via the address counter to the P register in a manner similar to that described before in connection with the skip timing diagrams. However, there is one difference in that PAC ADD **FF7–112** is in its clear condition **at** MP15 time of machine cycle **75**. This is so, inasmuch as the set condition of the Extend Repeat **FF41–10** prevents PAC ADD **FF7–112** from being set at MP9 time even though a Skip Op Code 10 is in **U0–5**. Therefore, the **NI43** address in P* register is only incremented by 1 so that P register now contains the address of **NI44**. Therefore, it will be seen that **NI43** is neither acquired nor executed following termination of the repeat operation.

In addition to the above functions, the detection of the skip condition also causes **Ā40–21** to clear **U0–9** so that there can be no operand reference of the **CI42** "T" address portion during machine cycle **76**. Thus, P register will always be able to reference memory for **NI44**. In addition, **Ā40–39** clears Next Instruction **FF41–12** to enable the generation of the signal P To W, as well as to insure that U register will be cleared at MP9 time of machine cycle **76**. At MP0 time of machine cycle **76**, **Ā41–22** clears the Repeat Mode **FF41–10** which thereby prevents Normal Advance PAC **FF39–31** from being set at MP2 time of machine cycle **76**. The address of **NI44** in P register may consequently be incremented by 1 during machine cycle **76** to prepare it for the acquisition of **NI45** during machine cycle **77**. Furthermore, during MP1 through MP3 times of machine cycle **76**, the R To OR Gates **FF18–03** is set to return the content of R register back to R memory, instead of returning the contents of **U*10–23** thereto as is normally done during the repeat operation. Therefore, the first timing diagram of FIGURE 54 illustrates the operation when a skip condition is detected during a machine cycle in which the content of R memory location 7 does not equal —1.

The center timing diagram of FIGURE 54 illustrates the function performed by the programming system when a skip condition is detected in a machine cycle during which the R memory location 7 number is equal to —1. As will be recalled in connection with FIGURE 53, the sensing of a —1 in R memory location 7 indicates that only one more execution of **NI42** need be accomplished. However, the detection of a skip condition in machine cycle **76** indicates that this final execution during machine cycle **77** need not be performed. As noted in FIGURE 53 and again in FIGURE 54, the sensing of a —1 in R register during machine cycle **76** causes **Ā41–18** to transfer the contents of P* register to P at MP13 time, such that the latter now contains the address of **NI43**. However, the subsequent sensing at MP14 time of the skip condition clears P register of **NI43** so that **Ā40–38** at MP15 time transfers P* via the address counter to P register. Since PAC ADD **FF7–112** is cleared at this time, 1 is added to the **NI43** address of P* such that P register now holds **NI44** address. Therefore, it is seen that the operation of the system in machine cycle **76** in the center diagram of FIGURE 54 is quite similar to the operation of the system in machine cycle **76** of the leftmost timing diagram. During machine cycle **77**, therefore, P register references the permanent memory for **NI44** which will thereafter be executed during machine cycle **78**.

The third situation in which a skip condition may be sensed is that occurring during the final execution of **NI42**, as determined by the fact that R register content=—1 during the preceding machine cycle. For example, in FIGURE 54, in the right most timing diagram, assume that R register equals —1 during the preceding machine cycle **76**, as is more clearly shown in FIGURE 53. In this case, during machine cycle **77** the **CI42** "T" address portion now residing in **U*10–23** references the memory to extract the operand required for the final execution of the repeated Skip instruction. It will also be seen from FIGURE 53 that during machine cycle **76 P** register has placed therein the address of **NI43** so that it is prepared to reference a memory for this instruction during machine cycle **77** unless overlap is inhibited. Assume now that the **CI42** "T" address portion in machine cycle **77** references permanent memory as does P register, so that **NI43** cannot be acquired during the final execution of the Skip instruction. In this event, the Normal Advance PAC **FF39–31** is set at MP2 time by **Ā39–30** so that P register cannot be incremented by 1 during machine cycle **77**. Thus, both P and P* registers hold the **NI43** address at this time. Also during this cycle, the **CI42** operand is withdrawn from permanent memory and sent to the arithmetic unit to determine the equality or non-equality between it and the accumulator contents. Now assume that equality is detected during this final execution, which is indicated at MP14 time of machine cycle **77** by the enabling of **Ā40–31** to set Skip **FF40–35**. Since the Normal Advance PAC **FF39–31** is also set at this time, **Ā40–37** clears P register of its **NI43** address so that P* can be subsequently gated through the address counter to place the **NI44** address in P register by MP15 time of machine cycle **77**. PAC ADD Flip-Flop **7–112** is in its clear condition at this time because Extended Repeat **FF41–10** remains set during machine cycle **77** and is not cleared until MP13 time. This is almost one machine cycle later than the clearing of Repeat Mode **FF41–10**. Therefore, the purpose of providing the Extend Repeat **FF41–10** is seen inasmuch as it is necessary to prevent the incrementing of the P* register content by a count of 2 when P* register itself holds the address of the instruction to be skipped.

FIGURE 55 is a timing diagram showing the operation of the invention in the event that an Interrupt or a Data Input-Output request is made during the execution of a repeated instruction. In this event, the execution of the repeated instruction is interrupted and the Interrupt sub-routine or the data transfer performed, after which the repeated instruction is again executed until normal termination occurs. Referring now to both FIGURE 53 and FIGURE 55, assume that a request Error Interrupt has been made of the system during machine cycle **74** so that flip-flop **42–12** is set by the commencement of machine cycle **75**. If no higher priority exists, then **Ā42–24** is enabled to set Priority 3 **FF42–18** at MP1 time. At MP1, **Ā39–60** also transfers **U*10–23** to the V register so that the variable memory is referenced for the operand to be used in the first execution of the repeated instruction NI42. At MP2 of machine cycle **75**, Ā39–32 is responsive to either one of the signals Disable Advance PAC from FIGURE 41 or Data Transfer/Interrupt from FIGURE 42 to set Normal Advance PAC Lockout **FF39–31**. Also at MP2, the presence of the Data Transfer/Interrupt signal at **Ā41–33** enables same to set Interrupt Repeat FF41–32 and clear Next Instruction Lockout FF41–12. The clearing of **FF41–12** permits **Ā41–24** to clear U register at MP9 time of machine cycle **75** in preparation for entry of the forced Interrupt instruction which is generated because of the set condition of Priority 3 **FF42–18**.

As during machine cycle **75** in FIGURE 53, **Ā14–33** forces a decimal 7 into U*6–9 so that R memory location 7 contents are withdrawn and placed in R register by MP5 time. It is here assumed that these contents do not equal −1 so that further executions of CI**42** must be accomplished before the repeat operation is considered terminated. At MP9 time, **Ā18–95** sets R To OR Gates **FF18–93** to return the R register contents to R memory. However, the set condition of Interrupt Repeat **FF41–32** prevents enabling of **Ā41–14** so that R Memory ADD FF remains clear during machine cycle **75** of FIGURE 55. Therefore, the R register is not incremented by 1 as is normally the case in the repeat operation. At MP9, **Ā42–32** clears U register. At MP10 time the Interrupt Op Code **46** is forced into **U0–5** and decimal address 15 and "S" address 10000 are respectively forced into **U6–9** and **U10–23** by the circuitry of FIGURE 43 in the manner previously described in connection with the interrupt operation. At MP13, R memory is referenced according to the "J" address 15 held now by **U*6–9**. At MP15 time of machine cycle **75**, **Ā16–35** clears R register. At MP**0** time of the next following machine cycle **Ā16–25** transfers the content of P register into R register so that the latter now contains the address of **NI41** which in turn is the Repeat instruction 40 previously acquired from memory during machine cycle **73** of FIGURE 53. Thereafter, the setting of **FF18–93** transfers the NI41 address from R register back into **T** memory location 15 in which it will be held until termination of the interrupt sub-routine. Upon termination of the interrupt sub-routine, the last instruction thereof will cause a reference of R memory at location 15 to withdraw the **NI41** address for use in referencing permanent memory to again acquire the Repeat instruction 40 therefrom. The system will thereafter execute **NI41** as done in machine cycle of FIGURE 53 so as to again enable the next following instruction **NI42** to be repeated a number of times specified by the number held in R memory location 7. Thus, the grant of an interrupt during execution of a repeated instruction merely discontinues the repeat operation until after the interrupt sub-routine has been performed.

Returning now to machine cycle **76** of FIGURE 3, the error "S" address portion now residing in **U*10–23** is used at **MP1** time to reference the permanent memory for the first instruction of the interrupt sub-routine identified as **NI85** in the timing diagram. Thereafter, P register is cleared by **Ā5–31** at MP2 time so that U*10–23 is transferred thereto by **Ā5–25** at MP3 time. P register now contains the address of NI**85** which will subsequently be incremented by 1 during machine cycle **76** in the usual fashion due to the cleared condition of Normal Advance PAC **FF39–31**. Therefore, Ā gates **39–36** through **39—39** are enabled successively so that the address residing in P register by MP6 time of machine cycle **76** is that of NI**86**, the second instruction of the interrupt sub-routine program.

As shown in FIGURE 41, the setting of Interrupt Repeat FF41–32 during a machine cycle prevents **Ā41–17** from setting the Terminate Repeat **FF41–16** even if the contents of R memory location 7 are equal to −1 at this time. As will be recalled from the discussion of FIGURE 53, a R register content of −1 indicates that one more execution of the repeated instruction must be performed in the next following machine cycle. However, when an interrupt request is granted during the machine cycle in which R register equals −1, the repeat operation must be discontinued at once. Therefore, the setting of Terminate Repeat **RR41–16** must be prevented during the cycle in which the next to last execution of the repeated instruction occurs, otherwise the signal P* To P from **Ō41–19** would replace the address of the Repeat instruction 40 in P register with the address in P* register prior to the time that P is transferred into R memory.

4.8. *Illegal values*

If during any of the above described operations illustrated by FIGURES 45 through 55, an illegal address in P register or an illegal operation code in **U0–5** is detected by circuitry previously described, then the overlap operation is modified as follows. In the former situation, the signal from **Ā5–50** (in absence of a required instruction jump) sets Normal Advance PAC Lockout **FF39–31** to prevent a change in the P register contents. The illegal address resides in P register thereafter until the error diagnostic circuits inform the operator of this fact, after which P register may be manually cleared by means not shown in FIGURE 5. However, in the event that an illegal operation code is detected in **U0–5**, **Ā8–81** is enabled at MP12 time of the cycle in which the instruction is acquired to transfer the memory address of the illegal instruction from **P*** register to P register so that this memory location may again be referenced during the next following machine cycle. If, after acquiring this instruction for the second time, there is no error indicated because of an illegal operation code, the system resumes its normal operation. However, if an illegal operation code is still sensed for this instruction, such will again be indicated to the error checking circuitry of the system for it to take appropriate action not forming a part of the present invention.

If U***10–23** contains certain memory addresses, then the overlap procedure during the above described operations is also modified as follows. A **U*** address in the range of from 34000 through 37717 is considered illegal as regards the referencing of a permanent memory for an operand data word. Consequently, a significant negative output from **Ā39–65** inhibits **Ā35–15** from setting **W** Operand **FF39–10**. It is also impossible to acquire instructions from the memory addresses from 34000 through 37777, since the twistor input switch is not used to store instructions. During the execution of a jump instruction, the **U*10–23** stages contain a memory address from whence is to be acquired the next instruction. Therefore, an output from **Ō39–74** disables **Ā39–19** from setting W Instruction **FF39–13** at this time. In addition, an output from **Ā39–82** signifies that **U*10–23** contains an instruction address for either the permanent or variable memories which is illegal. This output is transferred via **Ō39–33** to set **FF39–31** which prevents the incrementing of this illegal instruction address after it has been transferred from **U*10–23** to the P register. Therefore, since the illegal address thereafter is maintained in P register, the Illegal P Address Error circuits operate in FIGURE 5 as above described to set **FF39–31** in each machine cycle until the P register is cleared manually.

Although a preferred embodiment of the invention has been shown and described, it is apparent that many modifications may be made thereto by one skilled in the art without departing from the spirit of the invention as defined in the appended claims.

We claim:

1. In a stored program data processing system operating in successive machine cycles and utilizing words of data and words of instruction, the latter having a format including a word memory address portion and an operation code portion which may require a memory reference during execution, the combination comprising: a plurality of independent memories each having access circuits individual thereto and each storing both intsruction and data words at address locations therein, first means for generating the memory address of an instruction word, second means for holding an instruction word for command translation, third means using said generated address from said first means during a machine cycle to normally reference one of said memories in order to acquire an instruction word therefrom and transfer it to said second means, fourth means responsive to a memory reference code of an instruction word held in said second means for using its associated word memory address portion during the machine cycle immediately following that in which said last named instruction was acquired to reference one of said memories, fifth means inhibiting said third means from referencing a memory during a machine cycle when said fourth means references the same memory during the same machine cycle, and sixth means responsive to the operation of said third means for causing said first means to generate the memory address of the next instruction word in the program.

2. The combination according to claim **1** wherein said plurality of independent memories comprises two independent memories.

3. The combination according to claim **1** where said first means normally generates sequential memory addresses.

4. The combination according to claim **3** wherein said plurality of independent memories comprises two independent memories.

5. The combination according to claim **1** which further includes means responsive to a certain operation code value held by said second means for clearing same therefrom during the machine cycle in which it is acquired and for forcing said first means to again generate the memory address of the instruction word of which said operation code value forms a portion.

6. The combination according to claim **5** wherein said plurality of independent memories comprises two independent memories.

7. The combination according to claim **1** wherein said first means includes a first register for holding an instruction word memory address, a second register for holding an instruction word memory address, first means for transferring the address in said first register to said second register, and means to change the address in said first register.

8. The combination according to claim **7** wherein said plurality of independent memories comprises two independent memories.

9. The combination according to claim **7** wherein said address changing means includes a modifying unit and second gating means for transferring the address in said second register through said modifying unit to said first register.

10. The combination according to claim **9** where said modifying unit is normally enabled to add 1 to the address from said second register.

11. The combination according to claim **9** wherein said third means uses only the address in said first register.

12. The combination according to claim **7** wherein said third means uses only the address in said first register, and which further includes second gating means for transferring the address in said second register to said first register.

13. The combination according to claim **12** wherein said plurality independent memories comprises two independent memories.

14. The combination according to claim **12** wherein said address changing means includes a modifying unit and third gating means for transferring the address in said second register through said modifying unit to said first register.

15. The combination according to claim **14** wherein said modifying unit is enabled to add 1 to the address from said second register.

16. The combination according to claim **1** wherein said first means includes a first register for holding the instruction word memory address used by said third means, a second register for holding an instruction word address, first gating means for transferring the address fom said first register to said second register, a modifying unit for adding a value of N to an address, and second gating means for transferring the address from said second register through said modifying unit to said register, with said sixth means causing said first and second gating means to operate in the sequence.

17. The combination according to claim **16** wherein said plurality of independent memories comprises two independent memories.

18. The combination according to claim **16** where said modifying unit is enabled to add 1 to the address from said second register.

19. The combination according to claim **1** wherein said second means includes a first register for holding an instruction word and a second register for holding an instruction word.

20. The combination according to claim **19** wherein there are operation code command translator circuits individual to each of said first and second registers.

21. The combination according to claim **1** wherein said second means includes a first register for holding an instruction word transferred to it during a machine cycle by said third means, a second register for holding an instruction word whose associated word memory address portion is used by said fourth means, and first gating means for transferring a word in said first register to said second register.

22. The combination according to claim **21** wherein said plurality of independent memories comprises two independent memories.

23. The combination according to claim **21** wherein there are operation code command translator circuits individual to each of said first and second registers.

24. The combination according to claim **21** wherein said first means includes a third register for holding the instruction word memory address used by said third means, a fourth register for holding an instruction word memory address, second gating for transferring the address from said third register to said fourth register, a modifying unit of adding a value of N to an address, and third gating means for transferring the address from said fourth register through said modifying unit to said third register, with said sixth means causing said second and third gating means to operate in this sequence.

25. The combination according to claim **24** wherein said plurality of independent memories comprises two independent memories.

26. The combination according to claim **24** where said modifying unit is enabled to add 1 to the address from said fourth register address.

27. In a stored program data processing system operating in successive machine cycles and utilizing words of data and words of both arithmetic and non-arithmetic instructions, the instruction words having a format including a word memory address portion and an operation code portion which may require a memory reference during execution, where said system includes an arithmetic unit in which some arithmetic instructions require only one machine cycle for execution while other arithmetic instructions require an extended sequence of more than one machine cycle for execution, the combination comprising: a plurality of independent memories each having access circuits individual thereto and each storing both instruction and data words at address locations therein, first means for generating a memory address of an instruction word, second means for holding an instruction word for command translation, third means using said generated address from said first means during a machine cycle to normally reference one of said memories in order to acquire an instruction word therefrom and transfer it to said second means, fourth means responsive to a memory reference code of an instruction word held in said second means for using its associated word memory address portion during the machine cycle immediately following that in which said last named instruction is acquired to reference one of said memories, fifth means inhibiting said third means from referencing a memory during a machine cycle when said fourth means references the same memory during the same machine cycle, sixth means responsive to the operation of said third means for causing said first means to generate the memory address of the next instruction in the program, and seventh means for clearing said second means, during any machine cycle of an extended sequence execution except the final machine cycle thereof, of at least the operation code of any arithmetic instruction word acquired therein and for forcing said first means to again generate the memory address of the arithmetic instruction word of which said cleared operation code forms a portion.

28. The combination according to claim **27** wherein said plurality of independent memories comprises two independent memories.

29. The combination according to claim **27** wherein said first means includes a first register for holding the instruction word memory address used by said third means, a second register for holding an instruction word memory address, first gating means for transferring the address from said first register to said second register, a modifying unit for adding a value of N to an address, and second gating means for transferring the address from said second register through said modifying unit to said first register, with said sixth means causing said first and second gating means to operate in this sequence.

30. The combination according to claim **29** wherein said plurality of independent memories comprises two independent memories.

31. The combination according to claim **29** where said modifying unit is enabled to add 1 to the address from said second register.

32. The combination according to claim **29** wherein said first means further includes third gating means for transferring the address from said second register to said first register, with said seventh means causing said third gating means to operate.

33. The combination according to claim **32** wherein said plurality of independent memories comprises two independent memories.

34. The combination according to claim **33** where said modifying unit is enabled to add 1 to the address from said second register.

35. The combination according to claim **27** wherein said second means includes a first register for holding an instruction word transferred to it during a machine cycle by said third means, a second register for holding an instruction word whose associated word memory address portion is used by said fourth means, and first gating means for transferring a word in said first register to said second register.

36. The combination according to claim **35** wherein said plurality of independent memories comprises two independent memories.

37. The combination according to claim **35** wherein there are operation code command translator circuits individual to each of said first and second registers.

38. The combination according to claim **35** wherein said first means includes a third register for holding the instruction word memory address used by said third means, a fourth register for holding an instruction word address, second gating means for transferring the address from said third register to said fourth register, a modifying unit for adding a value of N to an address, and third gating means for transferring the address from said fourth register through said modifying unit to said second register, with said sixth means causing said second and third gating means to operate in this sequence.

39. The combination according to claim **38** wherein said first means further includes fourth gating means for transferring the address from said fourth register to said third register, with said seventh means causing said fourth gating means to operate.

40. The combination according to claim **39** wherein said plurality independent memories comprises two independent memories.

41. In a stored program data processing system operating in successive machine cycles and utilizing words of data and words of instruction, the latter having a format including a word memory address portion and an operation code portion which may require a memory reference during execution, the combination comprising a plurality of independent memories each having access circuits individual thereto and each storing both instruction and data words at address locations therein, first means for generating a memory address of an instruction word, second means for holding an instruction word for command translation, third means using said generated address from said first means during a machine cycle to normally reference one of said memories in order to acquire an instruction word therefrom and transfer it to said second means, fourth means responsive to a memory reference code of an instruction word held in said second means for using its associated word memory address portion during the machine cycle immediately following that in which said last named instruction is acquired to reference one of said memories, fifth means inhibiting said third means from referencing a memory during a machine cycle when said fourth means references the same memory during the same machine cycle, sixth means responsive to the operation of said third means for causing said first means to generate the memory address of the next instruction in the program, seventh means responsive to a memory reference jump operation code of an instruction word held in said second means during the machine cycle immediately following that in which it is acquired to inhibit operation of said third means, and eighth means responsive to the operation of said seventh means during said last named machine cycle for transferring the word address portion of said jump operation code instruction to said first means and thereafter causing said sixth means to operate.

42. The combination according to claim **41** wherein said plurality of independent memories comprises two independent memories.

43. The combination according to claim **41** wherein said first means includes a first register for holding the instruction word memory address used by said third means, a second register for holding an instruction word address, first gating means for transferring the addresss from said first register to said second register, a modifying unit for adding a value of N to an addresss, and second gating means for transferring the address from said register through said modifying unit to said first register, with said sixth means causing said first and second gating means to operate in this sequence.

44. The combination according to claim **43** wherein said plurality of independent memories comprises two independent memories.

45. The combination according to claim **41** wherein said second means includes a first register for holding an instruction word transferred to it during a machine cycle by said third means, a second register for holding an instruction word whose associated word memory address portion is used by said fourth means, and first gating means for transferring a word in said first register to said second register.

46. The combination according to claim **45** wherein said plurality of independent memories comprises two independent memories.

47. The combination according to claim 45 wherein there are operation code command translator circuits individual to each of said first and second registers.

48. The combination according to claim 45 wherein said first means includes a third register for holding the instruction word memory address used by said third means, a fourth register for holding an instruction word address, second gating means for transferring the address from said third register to said fourth register, a modifying unit for adding a value of N to an address, and third gating means for transferring the address from said fourth register through said modifying unit to said third register, with said sixth means causing said second and third gating means to operate in this sequence.

49. The combination according to claim 48 wherein said plurality of independent memories comprises two independent memories.

50. In a stored program data processing system operating in successive machine cycles and utilizing words of data and words of instruction, the latter having a format including a word memory address portion and an operation code portion which may require a memory reference during execution, and/or a skip condition test during execution to determine if the next instruction word in the program is to be executed, the combination comprising: a plurality of independent memories each having access circuits individual thereto and each storing both instruction and data words at address locations therein, first means for generating a memory address of an instruction word, second means for holding an instruction word for command translation, third means using said generated address from said first means during a machine cycle to normally reference one of said memories in order to acquire an instruction word therefrom and transfer it to said second means, fourth means responsive to a memory reference code of an instruction word held in said second means for using its associated word memory address portion during the machine cycle immediately following that in which said last named instruction is acquired to reference one of said memories, fifth means inhibiting said third means from referencing a memory during a machine cycle when said fourth means references the same memory during the same machine cycle, sixth means responsive to the operation of said third means for causing said first means to generate the memory address of the next instruction in the program, and seventh means responsive to the determination of a skip condition during execution in a machine cycle of a skip instruction word held in said second means for preventing execution of the instruction word acquired during said last named machine cycle and for insuring that said first means generates the memory address of the instruction word in the program which follows the instruction word whose execution is prevented.

51. The combination according to claim 50 wherein said plurality of independent memories comprises two independent memories.

52. The combination according to claim 50 wherein said seventh means operates to clear said second means of at least the operation code portion of said instruction word whose execution is to be presented.

53. The combination according to claim 52 wherein said plurality of independent memories comprises two independent memories.

54. The combination according to claim 50 wherein said first means includes a first register for holding the instruction word memory address used by said third means, a second register for holding an instruction word memory address, first gating means for transferring the address from said first register to said second register, a modifying unit for adding a value of N to an address, and second gating means for transferring the address from said second register through said modifying unit to said first register, with said sixth means causing said first and second gating means to operate in this sequence.

55. The combination according to claim 54 wherein said plurality of independent memories comprises two independent memories.

56. The combination according to claim 54 wherein said modifying unit can be selectively enabled to add a value of M to an address, with said seventh means being effective, if said third means is inhibited, to operate said second gating means and to enable said modifying unit for adding value M.

57. The combination according to claim 56 wherein said plurality of independent memories comprises two independent memories.

58. The combination according to claim 56 wherein value N is equal to 1 and value M is equal to 2.

59. The combination according to claim 58 wherein said plurality of independent memories comprises two independent memories.

60. The combination according to claim 50 wherein said second means includes a first register for holding an instruction word transferred to it during a machine cycle by said third means, a second register for holding an instruction word whose associated word memory address portion is used by said fourth means, and first gating means for transferring a word in said first register to said second register.

61. The combination according to claim 60 wherein said plurality of independent memories comprises two independent memories.

62. The combination according to claim 60 wherein said seventh means operates to clear the operation code portion of said first register.

63. The combination according to claim 62 wherein said plurality of independent memories comprises two independent memories.

64. The combination according to claim 62 wherein said first means includes a third register for holding the instruction word memory address used by said third means, a fourth register for holding an instruction word address, second gating means for transferring the address from said third register to said fourth register, a modifying unit for adding a value of N to an addres, and third gating means for transferring the address from said fourth register through said modifying unit to said third register, with said sixth means causing said second and third gating means to operate in this sequence.

65. The combination according to claim 64 wherein said modifying unit can be selectively enabled to add a value of M to an address, with said seventh means being effective, if said third means is inhibited, to operate said third gating means and to enable said modifying unit for adding value M.

66. The combination according to claim 65 wherein said plurality of independent memories comprises two independent memories.

67. The combination according to claim 66 where value N is equal to 1 and value M is equal to 2.

68. In a stored program data processing system operating in successive machine cycles and utilizing words of data and words of instruction, the latter have a format including a word memory address portion and an operation code portion which may require a memory reference during execution, or the repeated excution of the next following program instruction, the combination comprising: a plurality of independent memories each having access circuits individual thereto and each storing both instruction and data words at address locations therein, second means for holding an instruction word for command translation, third means using said generated address from said first means during a machine cycle to normally reference one of said memories in order to acquire an instruction word therefrom and transfer it to said second means, fourth means responsive to a memory reference code of an instruction word held in said second means for using its associated word memory address portion during the machine cycle immediately following that in which said

last named instruction is acquired to reference one of said memories, fifth means inhibiting said third means from referencing a memory during a machine cycle when said fourth means references the same memory during the same machine cycle, sixth means responsive to the operation of said third means for causing said first means to generate the memory address of the next instruction in the program, and seventh means responsive to a repeat operation code of an instruction word held in said second means for inhibiting operation of said third means for a time beginning with the machine cycle following that in which the repeated instruction is acquired and ending by commencement of the machine cycle during which occurs the final execution of the repeated instruction.

69. The combination according to claim **68** wherein said plurality of independent memories comprises two independent memories.

70. The combination according to claim **68** wherein said first means includes a first register for holding the instruction word memory address used by said third means, a second register for holding an instruction word memory address, first gating means for transferring the address from said first register to said second register, a modifying unit for adding a value of N to an address, and second gating means for transferring the address from said second register through said modifying unit to said first register, with said sixth means causing said first and second gating means to operate in this sequence.

71. The combination according to claim **70** wherein said plurality of independent memories comprises two independent memories.

72. The combination according to claim **68** wherein said second means includes a first register for holding an instruction word transferred to it during a machine cycle by said third means, a second register for holding an instruction word whose associated word memory address portion is used by said fourth means, and first gating means for transferring a word in said first register to said second register.

73. The combination according to claim **72** wherein said plurality of independent memories comprises two independent memories.

74. The combination according to claim **72** wherein said first means includes a third register for holding the instruction word memory address used by said third means, a fourth register for holding an instruction word address, second gating means for transferring the address from said third register to said fourth register, a modifying unit for adding a value of N to an address, and third gating means for transferring the address from said third register through said modifying unit to said third register, with said sixth means causing said second and third gating means to operate in this sequence.

75. The combination according to claim **74** wherein said plurality of independent memories comprises two independent memories.

76. The combination according to claim **75** where said modifying unit is enabled to add 1 to the address from said second register.

77. In a stored program data processing system operating in successive machine cycles and utilizing words of data and words of instruction, the latter having a format including a word memory address portion and an operation code portion which may require a memory reference during execution, where means are present for requesting an interrupt in the processing of instructions in a program in order to execute a special sub routine or to transfer a word between an external unit and the system, the combination comprising: a plurality of independent memories each having access circuits individual thereto and each storing both instruction and data words at address locations therein, first means for generating a memory address of an instruction word, second means for holding an instruction word for command translation, third means using said generated address from said first means during a machine cycle to normally reference one of said memories in order to acquire an instruction word therefrom and transfer it to said second means, fourth means responsive to a memory reference code of an instruction word held in said second means for using its associated word memory address portion during the machine cycle immediately following that in which said last named instruction is acquired to reference one of said memories, fifth means inhibiting said third means from referencing a memory during a machine cycle when said fourth means references the same memory during the same machine cycle, sixth means responsive to the operation of said third means for causing said first means to generate the memory address of the next instruction in the program, and seventh means responsive to an interrupt request for inhibiting operation of said third means for one machine cycle.

78. The combination according to claim **77** wherein said plurality of independent memories comprises two independent memories.

79. The combination according to claim **77** which further includes eighth means responsive to said interrupt request for placing a special instruction word into said second means during the machine cycle in which said third means is inhibited by said seventh means.

80. The combination according to claim **79** wherein said plurality of independent memories comprises two independent memories.

81. The combination according to claim **77** wherein said first means includes a first register for holding the instruction word memory address used by said third means, a second register for holding an instruction word address, first gating means for transferring the address from said first register to said second register, a modifying unit for adding a value of N to an address, and a second gating means for transferring the address from said second register through said modifying unit to said first register, with said sixth means causing said first and second gating means to operate in this sequence.

82. The combination according to claim **81** wherein said plurality of independent memories comprises two independent memories.

83. The combination according to claim **79** wherein said first means includes a first register for holding the instruction word memory address used by said third means, a second register for holding an instruction word address, first gating means for transferring the address from said first register to said second register, a modifying unit for adding a value of N to an address, and second gating means for transferring the address from said second register through said modifying unit to said first register, with said sixth means causing said first and second gating means to operate in this sequence.

84. The combination according to claim **77** wherein said second means includes a first register for holding an instruction word transferred to it during a machine cycle by said third means, a second register for holding an instruction word whose associated word memory address portion is used by said fourth means, and first gating means for transferring a word in said first register to said second register.

85. The combination according to claim **84** wherein said plurality of independent memories comprises two independent memories.

86. The combination according to claim **84** which further includes eighth means responsive to said interrupt request for forcing a special word into said first instruction register during the machine cycle in which said third means is inhibited by said seventh means.

87. The combination according to claim **86** wherein said plurality of independent memories comprises two independent memories.

88. The combination according to claim **84** wherein said first means includes a third register for holding the instruction word memory address used by said third means, a fourth register for holding an instruction word address, second gating means for transferring the address from said third register to said fourth register, a modifying unit for adding a value of N to an address, and third gating means for transferring the address from said fourth register through said modifying unit to said third register, with said sixth means causing said second and third gating means to operate in this sequence.

89. The combination according to claim **86** wherein said first means includes a third register for holding the instruction word memory address used by said third means, a fourth register for holding an instruction word address, second gating means for transferring the address from said third register to said fourth register, a modifying unit for adding a value of N to an address, and third gating means for transferring the address from said fourth register through said modifying unit to said third register, with said sixth means causing said second and third gating means to operate in this sequence.

90. The combination according to claim **89** wherein said plurality of independent memories comprises two independent memories.

91. A control circuit which comprises: first means generating a first data manifestation which may lie within either of first or second data value ranges; second means generating a second data manifestation which may lie within either of said first or second data value ranges; third means connected with said first means for generating a first or second indication when said first data manifestation lies in said first or second data value ranges, respectively, fourth means connected with said second means for generating a third or fourth indication when said second data manifestation lies in said first or second data value ranges respectively; first, second, third and fourth bistable means each having a set and a clear state; fifth means generating a serial train of timing signals; sixth means transmitting one of said timing signals from said fifth means to each of said first, second, third and fourth bistable means for placing each in a clear state; seventh and eighth means each connected with said third means and each enabled by a second of said timing signals occurring later than said one timing signal for respectively responding to said first or second indications to respectively place said first or second bistable means in a set state; ninth means connected with said fourth means and enabled by another of said timing signals occurring later than said second timing signal for responding to said third indication to place said first and third bistable means in a clear and a set state, respectively; and tenth means connected with said fourth means and enabled by another of said timing signals occurring later than said second timing signal for responding to said fourth indication to place said second and fourth bistable means in a clear and a set state, respectively.

92. A control circuit according to claim **91** which further includes a fifth bistable means having a clear and a set state; eleventh means for placing said fifth bistable means in a clear state; twelfth means connected with both said third means and said ninth means for responding to said first indication in conjunction with the absence of said ninth means response to thereby place said fifth bistable means in a set state; and thirteenth means connected with both said third means and said tenth means for responding to said second indication in conjunction with the absence of said tenth means response to thereby place said fifth bistable means in a set state.

References Cited by the Examiner

UNITED STATES PATENTS

2,936,118	5/1960	Greene et al.	235—167
2,959,351	11/1960	Hamilton et al.	235—157 X
3,061,192	10/1962	Terzian	235—157
3,156,897	11/1964	Bahnsen et al.	340—172.5

OTHER REFERENCES

"Pilot" NBS Technical News Bulletin, pp. 26–28, February 1959.

ROBERT C. BAILEY, *Primary Examiner.*

MALCOLM A. MORRISON, *Examiner.*

D. M. ROSEN, W. M. BECKER, P. J. HENON,
Assistant Examiners.